NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

**CLASSIFYING PSTN SWITCHING STATIONS:
A NATIONAL SECURITY AGENCY APPLICATION**

by

Allen S. Olson

September 1998

Thesis Advisor:	R. Kevin Wood
Co-Advisor:	Norman D. Curet
Second Reader:	Gerald G. Brown

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	September 1998	Master's Thesis

4. TITLE AND SUBTITLE
CLASSIFYING PSTN SWITCHING STATIONS: A NATIONAL SECURITY AGENCY APPLICATION

5. FUNDING NUMBERS

6. AUTHOR(S)
Olson, Allen S.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
National Security Agency, Center for Operations Research, 9800 Savage Rd,
Fort Meade, MD 20755-6678

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)*

The U.S. National Security Agency wishes to predict the routing of messages over various communications networks. Before routing predictions can be made in a public switch telephone network (PSTN), the hierarchical level of the network's switching stations must be known. This thesis develops an integer linear programming model for accomplishing this classification. In this model, a PSTN is represented as a graph in which switching stations are nodes and the logical connections between the switching stations are arcs. Algebraic constraints represent the engineering standards common to PSTNs. The model also incorporates probabilistic inferences about the class of switching stations to improve classification accuracy for networks not following typical PSTN structural practices. Preprocessing routines that analyze the network's topology and employ various heuristics to reduce the size of the problem are evaluated. The model is implemented in GAMS Development Corporation's Generic Algebraic Modeling System and sample PSTNs are solved using IBM's Optimization Subroutine Library solver on a 166 MHz desktop personal computer. Accurate classification solutions are obtained in under 2 seconds for actual PSTNs, while extremely large notional networks of over 300 nodes and 900 arcs are solved in under 2 minutes.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Hierarchical PSTN, integer program, linear programming, telecommunications	109
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

Approved for public release; distribution is unlimited

CLASSIFYING PSTN SWITCHING STATIONS: A NATIONAL SECURITY AGENCY APPLICATION

Allen S. Olson
Major, United States Marine Corps
B.S., University of Minnesota, 1982

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
September 1998

Author: _____
Allen S. Olson

Approved by: _____
R. Kevin Wood, Thesis Advisor

_____
Gerald G. Brown, Second Reader

_____
Norman D. Curet, Co-advisor

_____
Richard E. Rosenthal, Chairman
Department of Operations Research

ABSTRACT

The U.S. National Security Agency wishes to predict the routing of messages over various communications networks. Before routing predictions can be made in a public switch telephone network (PSTN), the hierarchical level of the network's switching stations must be known. This thesis develops a integer linear programming model for accomplishing this classification. In this model, a PSTN is represented as a graph in which switching stations are nodes and the logical connections between the switching stations are arcs. Algebraic constraints represent the engineering standards common to PSTNs. The model also incorporates probabilistic inferences about the class of switching stations to improve classification accuracy for networks not following typical PSTN structural practices. Preprocessing routines that analyze the network's topology and employ various heuristics to reduce the size of the problem are evaluated. The model is implemented in GAMS Development Corporation's Generic Algebraic Modeling System and sample PSTNs are solved using IBM's Optimization Subroutine Library solver on a 166 MHz desktop personal computer. Accurate classification solutions are obtained in under 2 seconds for actual PSTNs, while extremely large notional networks of over 300 nodes and 900 arcs are solved in under 2 minutes.

THESIS DISCLAIMER

Specific computer code is not included in this thesis, although the programs developed in this research are available from the author. The reader is cautioned that these computer programs may not have been exercised for all cases of interest. While every effort has been made, within the time available, to ensure that the programs are free of computational and logic errors, they cannot be considered validated. Any application of these programs without additional verification is at the risk of the user.

TABLE OF CONTENTS

I. INTRODUCTION 1

A. PURPOSE 1
B. BACKGROUND 3

II. HIERARCHICAL ROUTING 7

A. CLASSES OF SWITCHING STATION 7
B. CALL ROUTING 8
C. NETWORK TOPOLOGIES 11

III. MODEL FORMULATION 13

A. MODEL ASSUMPTIONS 13
B. INDICES 15
C. DATA 15
D. VARIABLES 16
E. FORMULATION 17
F. HARD AND SOFT INFERENCES 19

IV. PERFORMANCE OF THE BASELINE MODEL 23

A. TEST NETWORKS 23
B. TESTING METHODOLOGY 26
C. PRELIMINARY TESTING 28
D. CONCLUSIONS FROM THE PRELIMINARY TESTS 36

V. PREPROCESSING 39

A. LEAF PLUCKING 39
B. RESTRICTING CLASSIFICATIONS AT THE LOWEST HIERARCHICAL LEVEL 40
C. LONGEST SHORTEST PATHS ANALYSIS 41
D. IDENTIFYING TOP-LEVEL NODES 44
E. REDUCING MODEL SIZE 46

VI. TESTING OF PREPROCESSING ROUTINES 47

A. BOUNDS ON VARIABLES 48
B. EQUATION REDUCTION 52
C. LOOPING ON ZCLASS 52

VII. TESTING SOFT INFERENCES 57

A. IMPACT OF SOFT INFERENCES ON SOLUTION TIME 58
B. ABILITY OF SOFT INFERENCES TO INFLUENCE THE SOLUTION 60
C. SOFT WEIGHTS FOR TOP-LEVEL NODES 62

VIII. CONCLUSIONS AND RECOMMENDATIONS 65

A. OPTIMAL FORMULATION 65
B. CONTRAST WITH THE INTELLIGENT ENUMERATION ALGORITHM 66
C. SHORTCOMINGS AND SUGGESTIONS FOR FURTHER RESEARCH 68

APPENDIX A. CHARACTERISTICS OF THE TEST NETWORKS 71

APPENDIX B. PRELIMINARY TESTING OF THE BASELINE FORMULATION 79

APPENDIX C. PARAMETER VALUES OF TOP LEVEL CANDIDATE NODES 83

APPENDIX D. RESULTS OF PREPROCESSING TESTS 85

LIST OF REFERENCES 89

INITIAL DISTRIBUTION LIST 91

LIST OF FIGURES

Figure 1. Example of a Hierarchical PSTN 4
Figure 2. Direct and Final Paths in Hierarchical Routing 10
Figure 3. A Route Table 10
Figure 4. Examples of Basic Network Topologies for PSTNs 11
Figure 5. Example of a Network with a Modified Longest Shortest Path 26
Figure 6. Range of the *TWT/PWT* Ratio Giving Correct Solutions 31
Figure 7. Effect on Solution Time as *ZWT* Varies 34
Figure 8. Effect on Solution Time as *TWT* Varies 35
Figure 9. Solution Speeds Attained During Selected Trials 37
Figure 10. Possible Configurations of L-S Paths 42
Figure 11. Effect of Preprocessing on the Relaxed Objective Function Value 50
Figure 12. Effect of Preprocessing on Solution Time 51
Figure 13. Improvement in Solution Time and Relaxed Objective Function Value 54
Figure 14. Cumulative solution times for the *Z_Loop* routine 56
Figure 15. Logical Structures of Test Networks 0 and 1 72
Figure 16. Logical Structures of Test Networks 2 and 3 73
Figure 17. Logical Structure of Test Network 4 74
Figure 18. Logical Structure of Test Network 5 75
Figure 19. Logical Structure of Test Network 6 76
Figure 20. Logical Structure of Test Network Tracy 77
Figure 21. Logical Structure of Test Network Balt 78

LIST OF TABLES

Table 1. Classes of Hierarchical Switching Stations 8
Table 2. Equation Reduction in the Model 53
Table 3. Soft Parameter Weights Used in Testing 57
Table 4. Accuracy of an Implementation of Soft Inference Rules. 59
Table 5. Model Behavior with the Introduction of Soft Inferences 59
Table 6. Scaling of Soft Inference Weights Yielding Alternate Solutions 60
Table 7. Solution Times of the Recommended Model 67
Table 8. Test Network Characteristics. 71
Table 9. Solution Speeds as *ZWT* and *TWT* are Varied 80
Table 10. Effect of Solver Options on Solution Times 81
Table 11. Effect of Branching Strategy on Solution Times 82
Table 12. Parameter Values of Top-level Candidate Nodes 83
Table 13. Gap Between the Relaxed and Optimal Objective Function Values 85
Table 14. Solution Times for Various Preprocessing Routines 86
Table 15. Solution Times for *Z_Loop* Strategy in the Baseline Formulation 87
Table 16. Solution Times for the *Z_Loop* Strategy with Additional Preprocessing 88

ACKNOWLEDGMENT

The author expresses his sincere gratitude for the assistance and guidance of Dr. Kevin Wood and Dr. Norm Curet. Whatever quality this thesis may possess results directly from their insight and many diligent, critical reviews. The author also greatly appreciates the magnanimous assistance of CDR John Brandeau, USN, who provided his extremely fast JAVA algorithm to be modified as the preprocessor used in this thesis work.

I. INTRODUCTION

Knowledge of the routes messages will take as they pass through a communications network can be exploited to enhance intelligence collection capabilities and evaluate network security. Accurately predicting the routes of telephone messages within a public switch telephone network (PSTN) is only possible when the hierarchical levels of the switching stations are known. Once the hierarchical levels of a PSTN's switches have been accurately classified, the network can be further processed to yield intelligence insights. This thesis presents an integer programming model that can infer the hierarchical levels of PSTN switching stations from the logical topology of a network, making best use of available information about the network to speed processing time and increase accuracy of the node classifications. The goal of this thesis is to develop this model and to evaluate it for suitability as a network analysis tool.

A. PURPOSE

The Department of Defense's National Security Agency/Central Security Service (NSA/CSS) has two national missions. The foreign signals intelligence (or SIGINT) mission requires the NSA/CSS to provide control and organization of all foreign signals collection and processing activities of the U.S. Government. The information systems security (INFOSEC) mission requires that NSA/CSS provide policy and services to aid in protecting U.S. information systems from exploitation (E.O. 12333, 1981).

Both missions of the NSA/CSS require good methods for predicting routes that messages will take over various communications network technologies. For the SIGINT mission, message-routing predictions would help focus collection efforts on high-payoff portions of target networks in adversary countries. Route prediction can

also support the INFOSEC mission by assessing areas of vulnerability to interception or unauthorized access of networks used by U.S. agencies.

The Generalized Communications Assessment Tool (GCAT) is a large-scale analysis tool under development for NSA/CSS to provide route prediction, and other analysis functions, over various communications technologies. In the fall of 1997, GCAT was incorporating methods for analysis of PSTNs. This thesis develops and evaluates an integer programming model (IP) for inclusion in the GCAT methods implementing PSTN route prediction. The IP will be evaluated primarily by its performance in classifying "hierarchical-routing" regional PSTNs from the United States, and modified versions of these PSTNs. A future goal is to extend the model to classify non-U.S. PSTNs.

In the model, a PSTN is represented as a graph in which the switching stations are nodes, and the trunk lines interconnecting the switching stations are arcs. Most of the world's telephone systems use a hierarchical-routing system, in which calls are referred to higher-level, more capable switches whenever needed to complete a connection. The model attempts to infer the hierarchical level of the switching stations by algebraically representing the network structure assumed in a hierarchy, and the engineering practices commonly observed in PSTNs. In some cases, PSTNs do not strictly follow the typical hierarchical structure, so the model can also incorporate inferences about the hierarchical level of switching stations.

GCAT is intended to be used interactively. Consequently, lengthy solution times for any of its component modules is undesirable. This thesis proposes and evaluates several routines for speeding solution time of the node classification IP. These routines reduce the dimensions of the problem prior to solving the model, dramatically reducing solution times.

B. BACKGROUND

GCAT's PSTN methods seek to generate route predictions by reverse-engineering the hierarchical structure of the network under study. The methods apply rules derived from PSTN routing protocols and standard engineering practices to surmise the functionality of the network. Some rules model hard engineering standards, while others are heuristics, true only some of the time.

1. Overview of Hierarchical PSTNs

Viewed globally, the public telephone system is an interconnected network of transmission media allowing virtually any telephone on earth to communicate with any other, more or less on demand. Certain structural conventions have been adopted in order to provide this service economically and with reasonable service performance to subscribers.

One such convention is the notion of a hierarchical structure. While less efficient than more recent dynamic routing technologies, hierarchical routing is still the most prevalent protocol worldwide (Ash, 1998). Hierarchical routing greatly reduces the requirement for complicated interactions between the switches of a network. This simplification was mandatory in order to construct telephone systems using the technology available in the early part of the 20th century.

Within a PSTN, calls are routed among interconnected switching stations, congestion permitting, so as to minimize the number of trunk lines used in the path (Noll, 1991). Calls that cannot be switched via shorter paths overflow onto less preferred paths, i.e., paths using more trunks. If no direct routing possibilities at a particular switch can complete the connection, the switch will, by default, route the

call to a higher-ranking switching station. The higher-ranking switch will have a wider geographic domain and increased ability to route calls traveling greater distances (Ash, 1998). An example of a hierarchical PSTN is depicted in Figure 1.

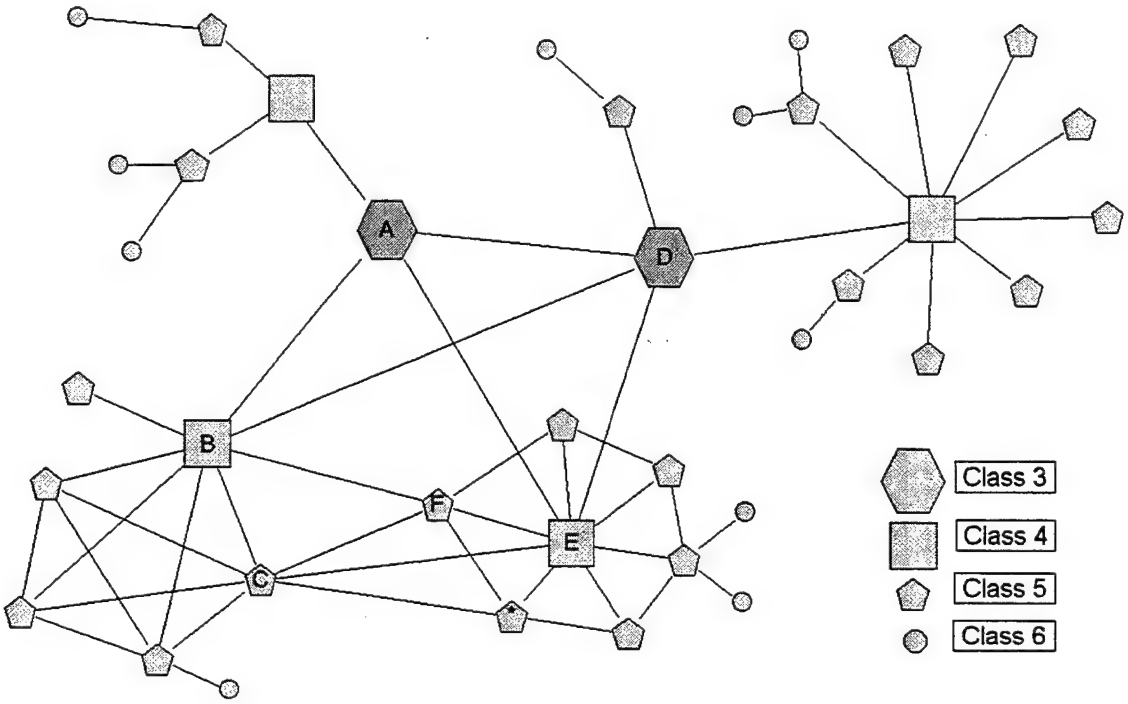

Figure 1. Example of a Hierarchical PSTN

This is an example of a "typical" hierarchical PSTN. Each node in the graph represents a switching station, while each arc indicates a path for routing telephone calls between the interconnected nodes. Hierarchical classes will be defined in greater detail later in the text; however, nodes with lower class numbers are higher in the PSTN's hierarchy, and more able to route calls travelling greater distances. The node annotations will be referred to later in the text.

2. Node Classification Using Artificial Intelligence

Prior to considering a mathematical programming approach to the PSTN node classification problem, a rules-based artificial intelligence (AI) routine for classification was tested and discarded. This precursor node-classification program was coded in NASA's C-Language Integrated Production System (CLIPS), a programming tool specialized for encapsulation of expert knowledge (Giarratano,

1997). Insights from this earlier effort can be adapted for use in the IP to increase classification accuracy for non-typical networks and to speed solution times.

The AI approach to the node classification problem attempts to capitalize on the conditional (usually, but not always, true) nature of many of the structural conventions of PSTNs. Telecommunications experts have provided heuristic rules that can be applied to the node-classification problem. For example, the type of equipment used at a switch facility can suggest the level of the station, and the commonality of a switch's operating company with others in the network can also provide clues to the level of the switch. Since the majority of these heuristic rules are true only some of the time, each in isolation can only suggest a likely classification for a node. Collectively, it was thought that these rules would enable an AI program to converge to an accurate hierarchical labeling of the switching stations.

Performance of the CLIPS node classification routine was unsatisfactory. It was slow to converge to a solution and had no clear stopping rule. The CLIPS routine was also a "black box"—the inner workings were obscure. There was no way to specify a partial solution, nor any way to tailor the algorithm for classification of networks known to vary from the norm. This motivated the development of alternative node-classification algorithms.

Currently, two complementary approaches to solving the node classification problem are under development at the Naval Postgraduate School. An "Intelligent Enumeration" algorithm is being developed that may be able to classify switching stations without resorting to solving an integer programming model. If this algorithm proves adaptable enough to cover the range of PSTNs studied with GCAT, its speedier solution times may make the algorithm a viable solution technique for the node-classification problem. The intelligent enumeration algorithm and the preprocessing routines of this thesis employ similar tactics in identifying critical features of the

network, and a version of the algorithm has been adapted to quickly accomplish the network preprocessing used in this thesis (Brandeau, 1998).

3. Contrasting the AI and Mathematical Programming Approaches

The mathematical programming model of this thesis differs from the AI approach in that the IP generates node classifications by first enforcing a baseline hierarchical structure. The workings of the IP model are analytically accessible, and the baseline model can be adapted in predictable ways. For example, certain countries or areas may exhibit a tendency to construct robust PSTNs with redundant routing, fewer hierarchical levels and proportionately more nodes at higher hierarchical levels (perhaps to improve resiliency when portions of the network sustain damage). In modeling these networks, the IP's parameters can be adjusted to solve for a network with fewer levels and more top-level nodes. With a basic network structure established, the IP incorporates some of the conditional rules of the AI module in order to improve classification accuracy on portions of a network not following hierarchical standards. Testing the efficacy of these so-called "soft inferences" in the IP is one of the goals of this thesis.

II. HIERARCHICAL ROUTING

Building a model of a hierarchical telephone system requires a deeper look at the classes of switches and the protocols used in routing calls. This chapter outlines the protocols and telecommunications practices that will later be used in developing an IP model.

A. CLASSES OF SWITCHING STATION

Functionally, there are two types of switching stations. Individual subscribers connect to the phone system via *local exchanges*, which are at the lowest hierarchical level. These exchanges can directly route traffic only between local customers. Calls between customers not of the same local exchange must be routed over trunk lines, often via *transit exchanges*. Transit exchanges are at the upper levels of the hierarchy, and switch only concentrated traffic destined for non-local destinations (Pearce, 1981).

Worldwide, there are two prevailing types of hierarchical PSTNs, namely, the ATT and CCITT protocols. Table 1 lists the various levels of switching station and their U.S. (ATT) and European (CCITT) nomenclature. In hierarchical PSTNs, each switching station except those of highest rank is subordinate to a higher level station that serves to concentrate traffic destined for regions beyond the geographic domain of the current level. In the ATT routing scheme, class 4 and lower-numbered switches are transit exchanges, routing concentrated traffic via trunk lines. Class 3 and 4 facilities are often referred to as *tandems*. End offices and remote concentrators, classes 5 and 6, connect individual subscribers to the network via *subscriber loops* (Freeman, 1989).

The ATT and CCITT protocols are quite similar, differing primarily in nomenclature and in that CCITT allows for seven hierarchical levels. In the ATT scheme, class 3 through 6 switches provide telephone service within a discrete

geographic region. It is within these regions that the IP attempts to classify nodes. Higher level switches exist (Regional and Sectional Centers), providing long-distance and international phone switching services at the national network level. GCAT will employ other methods to predict call routing at these levels, where hierarchical protocols are not used.

GCAT Nomenclature	ATT (North American)	CCITT (European)
Class 1	Regional Center	Quaternary Center
Class 2	Sectional Center	Tertiary Center
Class 3	Primary Center	Secondary Center
Class 4	Toll Center	Primary Center
Class 5	End Office	Local Office
* Class 6	Satellite, or Remote Concentrator	

Table 1. Classes of Hierarchical Switching Stations

The "class" of a hierarchical PSTN switching station refers to its level within the routing hierarchy. The lower a switch's class number, the greater its ability to route traffic travelling farther geographic distances. * Note: Remote concentrators do not represent a sixth hierarchical level, but in GCAT such facilities are considered "Class 6" exchanges.

B. CALL ROUTING

Hierarchical routing is particularly desirable for systems employing unsophisticated switches, as was the case when public telephone systems were first implemented. Hierarchical routing automatically ensures no call will be returned to a node previously used in the route (prevents looping), and also requires that connections be established using a reasonable number of trunk lines (Ash, 1998).

Physically, telephone calls travel via trunk lines, and the arrangement of these media (fiber, copper wire, etc.) is the *physical topology* of the network. The *logical topology* refers to how the nodes actually communicate. In the logical topology, interconnections (arcs) between switching nodes are called *links*. A link between two nodes i and j may be physically composed of several sets of trunks and intermediate switches; however, from the perspectives of switches i and j, a direct connection exists

between them. The hierarchical routing protocols described next operate within the context of a network's logical topology.

The set of paths available for routing calls between a pair of origin and destination nodes is referred to in GCAT as a *route table*. These paths are composed of two types of links: *direct* and *final*. *Direct links* may be established whenever an average high volume of traffic exists between any two nodes, regardless of the classes of the nodes. Direct links are essentially high-volume short-cuts. A node's *final link* connects it to its hierarchical parent. By following the final links from an originating office up through each hierarchical level, across (if necessary) to the destination node's predecessor parent at the top level, and then down via final links to the destination office, one would be tracing the *final path* (see Figure 2). The final path is formed of two routing *ladders*, one rising from the originating local exchange up to the top level, and another descending from the top level to the destination local office. In order to prevent any possibility of "call looping," the only valid routing paths between two local exchanges are those along the final path, or following direct links which short-cut the final path. In other words, paths routed through a node of a third hierarchical ladder are prohibited (Ash, 1998). Figure 2 shows several direct routing possibilities and the final path for an origin-destination pair of end offices.

While somewhat simplified, for purposes of this thesis the paths in a route table can be ordered by preference using two rules. Since call quality diminishes with increasing number of trunk lines used, paths using fewer links are preferred. It is also preferred that a switch advance a call as far as possible toward its destination. By this second rule, a switch will exhaust all direct routing possibilities at its level before defaulting and utilizing the final link to its parent switch higher in the hierarchy (Freeman, 1989). The final route is so called because it is the final opportunity to complete a call, since all direct routing possibilities will have been exhausted prior to

utilizing it. Figure 3 shows the route table generated by these rules for the example route of Figure 2. These paths are also depicted in Figure 1.

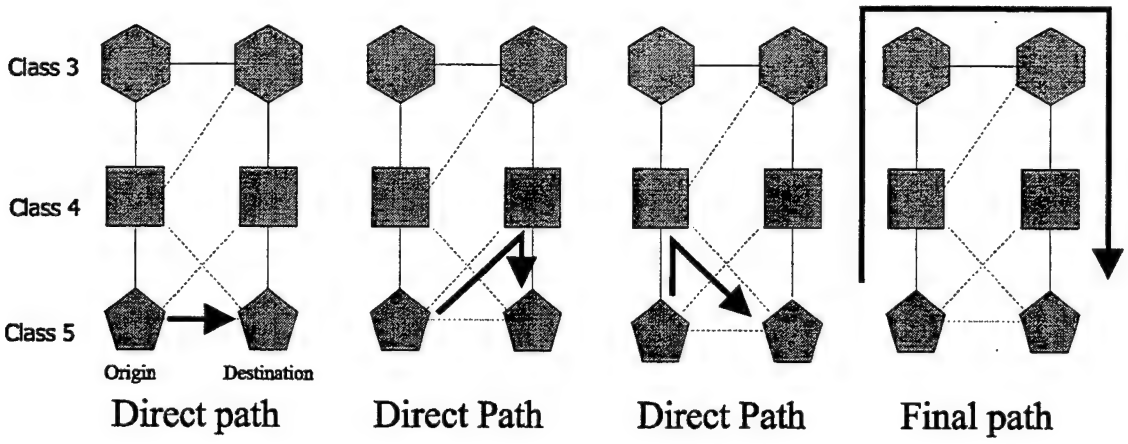

Figure 2. Direct and Final Paths in Hierarchical Routing

Final links are shown as solid lines, and direct links are dashed. This example identifies three of the four direct routing paths, and the final path. The fourth direct path would utilize the direct link between the class 4 and class 3 nodes.

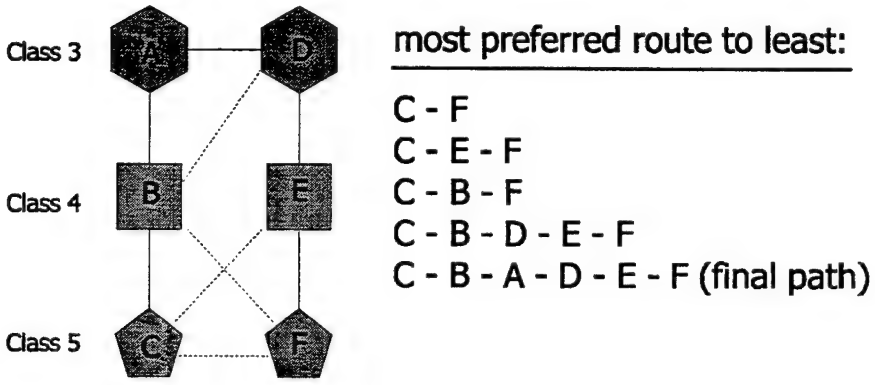

Figure 3. A Route Table

In a route table, more preferred paths use fewer trunks. Where this rule is ambiguous, the least preferred route uses a final link (indicated as solid lines) earlier in the path. Notice in Figure 1 that paths from C to F also exist through the node marked with an asterisk. These paths are invalid in the hierarchical routing protocol because a third ladder would be involved.

C. NETWORK TOPOLOGIES

The hierarchical routing scheme simplifies switching requirements, since only the default final route to a parent station, and the additional high-usage direct routes, need to be known by a switch in order to route calls (Ash, 1998). The issue then becomes one of configuring the network cost-effectively. There are four basic network configurations in general use: mesh, star, double star, and hub and spoke (see Figure 4). The configuration of the network has a major impact on solution time for the node-classification IP.

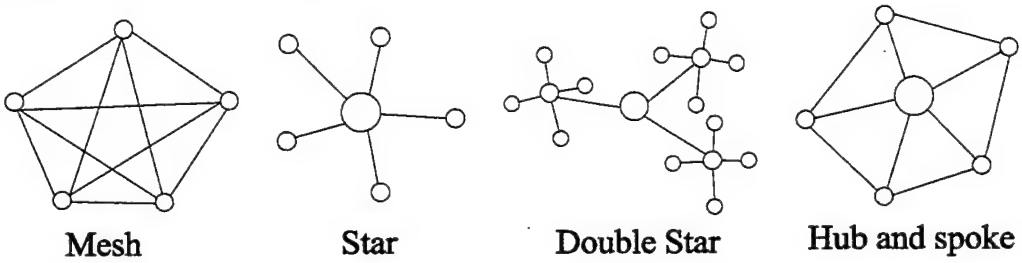

Figure 4. Examples of Basic Network Topologies for PSTNs

A regional PSTN may contain several of these topologies.

In a mesh-connected portion of a network, there are direct links between every pair of switches. This is a costly configuration indicating high traffic volumes between exchanges, such as in metropolitan areas. In a star configuration, every node is interconnected via a central exchange called a "tandem." Double-star configurations have satellite star networks interconnected via their tandems to higher-order tandems. Star configurations are typically found in lower traffic volume situations, such as rural areas. Hub and spoke formations are an intermediate configuration, offering some redundant routing possibilities without the expense of a full mesh (Freeman, 1989). Mesh, hub and spoke, and star configurations are also depicted as components of the example PSTN of Figure 1. There are no routing decisions to be made in star configurations, since every call is either local or passed to the tandem. Classifying the

hierarchical level of nodes in such a configuration is relatively simple. Networks with mesh, or hub and spoke, configurations are more difficult to classify, since there are many possible ways to assign hierarchical levels to the nodes.

III. MODEL FORMULATION

The chapter presents an integer programming model for classifying PSTN hierarchical levels, along with the assumptions underlying the model. The model seeks to be as general as possible; but the formulation is derived primarily from observations of U.S. (ATT) networks. Some of the assumptions, and their implementation in the model, may need to be revalidated for analysis of non-U.S. networks.

A. MODEL ASSUMPTIONS

From the description of hierarchical PSTN protocols in Chapter II, some assumptions can be drawn that will be used in the IP described in this chapter. In the interest of brevity later, each assumption is assigned a short-hand name.

- ***Final_Reqd***. Every node in the network is either subordinate to another node, or is at the top level of the network. Furthermore, a node will have at most one parent, and that parent will be at a higher hierarchical level. In telecommunications terms, every node not at the highest level will have a final link to its parent in the hierarchy.

- ***Top_Mesh***. Nodes at the top level of the network must form a complete (sub) graph (i.e., be completely interconnected). This is a requirement for the existence of route ladders between every pair of local exchanges.

Additionally, expert knowledge about typical telephone networks can be drawn upon to derive assumptions about the usual "shape" of PSTNs. Since these assumptions may not be universally true, they will appear in the IP model as "aspirations," rather than requirements.

- ***Min_Level***. A network will be constructed with the fewest possible number of hierarchical levels. The paths containing the most trunks, and therefore the most signal loss and inefficient trunk usage, will be the final paths defining the hierarchy. By reducing the number of hierarchical levels, the final paths will use the fewest possible trunks.

- ***Min_Tops***. The number of top-level nodes will be the minimum required to establish route ladders between all pairs of local exchanges. If functionality of the network does not require a top-level candidate to be at the top level, it is probably not a top-level node. This observation is most likely a result of economic incentives–it will be more economical to install direct trunks, whenever possible, rather than establishing a high-level switching facility.

- ***More_5s***. Class 5 end offices are the most common switches in a network. This is a logical result of the pyramid-shape typical of hierarchies. Since transit exchanges have increased geographic span of influence, fewer are required to span the domain. Class 6 remote concentrators are specialized entities, observed to be less common than end offices. Whenever a node may be one of several possible classes and still satisfy all other assumptions, most often the node will be a class 5 end office.

B. INDICES

Two indices are needed for the basic model. An additional, optional, subset will be described in section F.

- i – an element of the set of switching stations (nodes) of the network.

- c – an element of the set of possible switch classes. While this set can be generalized to represent arbitrary levels, in U.S. regional PSTNs the domain of this set is $\{3, 4, 5, 6\}$.

C. DATA

The basic input to the IP is the logical topology described by a node-node adjacency matrix (see Ahuja, et al, 1993, for a description of adjacency matrices). The adjacency matrix defines an undirected network $G = (N, A)$ with node set $N = \{1, 2,..., n\}$ and arc set $A = \{(i,j)\} \in N \times N$.

- $(i,j) \in A$ – an arc of the network; e.g. a logical link.

- *ZWT* – objective function weight whose relative proportion with other such scalars establishes the importance of the *Min_Level* assumption.

- *TWT* – objective function weight penalizing the number of top-level nodes in the solution. Implements the *Min_Tops* assumption.

- *PWT* – objective function weight rewarding the number of class 5 nodes in the solution. Implements the *More_5s* assumption.

- $SOFT_{ci}$ – soft inference parameter; an objective function weight applied to influence the class c assigned to node i in the final solution. Soft inferences are more completely described in a later section.

- $\underline{zclass}$ – minimum class allowable in the network. Imposes a lower bound on the lowest class used in the network.

- Δ – the difference between the highest and lowest possible hierarchical levels of the network. Defines the range of possible classes in the network.

D. VARIABLES

Four sets of variables are needed to represent the characteristics of PSTNs.

- *zclass* – an integer variable representing the minimum class used in the network. Given the inverse relationship between hierarchical level and the class number representing them, *zclass* is equal to the highest level used in the network.

- bcl_{ci} – a binary variable which is 1 if node i's class is c, and is 0 otherwise.

- top_i – a binary variable which is 1 if node i is at the top hierarchical level of the network, and is 0 otherwise.

- p_{ij} – a binary variable indicating if node j is the parent of node i. This is a surrogate for each node's final link.

E. FORMULATION

Maximize

$$ZWT \cdot zclass - TWT \cdot \sum_i top_i + PWT \cdot \sum_i bcl_{5i} + \sum_i \sum_c SOFT_{ci} \cdot (bcl_{ci}) \qquad \text{(obj)}$$

Subject to

$$\sum_c bcl_{ci} \qquad = 1 \quad \forall i \in N \qquad (1)$$

$$top_i + top_j \qquad \leq 1 \quad \forall (i,j) \notin A \qquad (2)$$

$$top_i + \sum_{j:(i,j)\in A} p_{ij} \qquad = 1 \quad \forall i \in N \qquad (3)$$

$$bcl_{ci} - \sum_{c':c'>c} bcl_{c'j} + p_{ji} \qquad \leq 1 \quad \forall c, (i,j) \in A \qquad (4)$$

$$p_{ij} + p_{ji} \qquad \leq 1 \quad \forall (i,j) \in A \qquad (5)$$

$$zclass - \sum_c (bcl_{ci} \cdot c) - top_i \qquad \leq -1 \quad \forall i \in N \qquad (6)$$

$$-zclass + \sum_c (bcl_{ci} \cdot c) + \Delta \cdot top_i \qquad \leq \Delta \quad \forall i \in N \qquad (7)$$

$$zclass \in \{\underline{zclass}, \ldots, \underline{zclass} + \Delta\}$$
$$bcl_{ci} \in \{0,1\} \ \forall c,i$$
$$top_i \in \{0,1\} \ \forall i$$
$$bcl_{ci} \in \{0,1\} \ \forall (i,j)$$

Constraints (1) require that every node be assigned a class.

Constraints (2) implement the *Top_Mesh* assumption by requiring that for every pair of nodes not connected by an arc, at most one may be a top-level node.

The *Final_Reqd* assumption is implemented by constraints (3) and (4). Each node must either be a top-level node, or must choose a parent. By (4), any parent must be at least one hierarchical level above its child. Notice (4) allows the possibility of a parent node being more than one hierarchical level above any children nodes.

Constraints (5) prevent nodes from being parents to each other. These constraints are logically redundant with (4), but adding them to the formulation speeds solution times (they are not redundant in the continuous linear program relaxation of the IP).

The last two constraints identify nodes eligible or not eligible to be tops. By constraint (6), nodes with binary class equivalent to *zclass* must be tops, while (7) requires that nodes with class greater than *zclass* not be tops. Collectively, constraints (6) and (7) require that *zclass* be equal to the smallest index *c* used in the network.

The remaining assumptions are implemented in the objective function (obj). The term containing *ZWT* rewards for fewer levels (the *Min_Level* assumption). The *TWT*-term penalizes the number of top-level nodes (*Min_Tops*), and the *PWT* term rewards for every class 5 node (*More_5s*). Note that in implementation, *PWT* may be absorbed into the $SOFT_{5i}$ data parameter. Choice of *ZWT*, *TWT* and *PWT* determine the relative importance of the *Min_Level*, *Min_Tops*, and *More_5s* assumptions, and when one assumption will overrule another.

Soft inferences are also implemented in the objective function. The *SOFT* terms reward for class assignments commensurate with those indicated by the soft data. Soft inference parameters can only be used when additional information about the network is available to invoke the heuristic rules generating them. In the absence of such data, the $SOFT_{ci}$ parameters are zero. The next section provides a full discussion of soft inferences.

F. HARD AND SOFT INFERENCES

Inferences are indications about the variable values based on intelligence data about the network, or originating from the analyst. This section describes the implementation of soft and hard inferences in the IP.

1. Hard Inferences

Hard inferences are input by the analyst and dictate a portion of the solution. This introduces the possibility of model infeasibility. Using hard inferences to specify some portion of the solution may be desired, for example, to conduct sensitivity analysis on the route tables under various assumptions about the class of a switch. Also, an analyst may surmise the network's actual configuration is not optimal given the model assumptions. The use of hard inferences will allow investigation of this possibility.

While the value of any variable of the model can be fixed as a hard inference, the model is optimized for analytical conjecture about the identity of top level nodes. Hard inferences can establish an additional subset and data parameter:

- $N_T \subset N$: A subset of N required by the analyst to be at the top hierarchical level of the network.

- *MINTOPS*: A data parameter establishing the minimum number of top-level nodes in the network.

To expedite the solver if $MINTOPS > 0$, an additional equation is added to the model, and the values of the top_i and p_{ij} fixed for all i in N_T:

$$\sum_i top_i \geq MINTOPS$$

$$top_i = 1 \qquad \forall i \in N_T$$

$$p_{ij} = 0 \qquad \forall i \in N_T$$

With the top_i variables either linearly constrained or fixed (which also requires that these nodes have no parents), the solver can take advantage of a partial solution. If the set N_T is empty and $MINTOPS = 0$, these portions of the model are inactive.

2. Soft Inferences

The purpose of soft inferences is to influence the formulation's solution to more correctly classify networks that do not entirely follow the model's assumptions. The premise behind soft inferences is that clues of a network's non-conformity may be found in various heuristic rules. This thesis implements four rules derived from the expert opinion of telecommunications analysts pertaining to U.S. regional networks. The purpose of soft inference testing in this thesis is to validate the methodology, not the rules specifically. Presumably, different rules would need to be developed for analysis of non-U.S. networks.

Soft inference parameters are generated for the appropriate classes of a node when a soft inference rule is invoked. In the objective function, these parameter weights encourage the solver to choose the class weighted by the soft parameter. The soft inference rules are cumulative. If several rules apply for a particular node, any soft parameters applied to the same class are summed. This tactic allows several weaker rules to cumulatively influence the class of a node more strongly than a single, stronger rule. The four rule sets used in later evaluation of soft inferences are described briefly below. The rules are named after telecommunications acronyms whose precise meanings are not pertinent to this thesis. It is expected that in some cases, data needed to employ similar rules may be available for non-U.S. PSTNs.

a) CLLI Rule

The premise behind the CLLI rule is that switches with large capacities are more likely to be transit exchanges (class 3 or 4 in the ATT scheme) than local exchanges (class 5 or 6). In ATT networks, a particular code associated with each switch (the "CLLI code") gives an indication of the switch's capacity. Codes ending in a "T" indicate a large capacity switch likely to be a tandem. When this condition is true for node i, the $SOFT_{ci}$ parameters for $c = 3$ and 4 are increased by an appropriate weighting factor.

b) NPACOC Rule

In North American networks, a code is available (the "NPACOC code") identifying the number of subscriber loops connected to a switching facility. If the code indicates there are no subscriber loops, the switch probably accomplishes trunk routing only, and is therefore unlikely to be a local exchange. When the condition for this rule is true for node i, the transit class *SOFT* parameters are increased by a weight associated with this rule.

c) OCN Rule

The Operating Company Name (OCN) rule identifies nodes that are unlikely to be tandems based on the commonality of the nodes' OCN with the most common OCN in the network. If the most common OCN of the network is known, and a node's OCN is also known and is not the most common, the node is more likely to be a local exchange than a transit exchange. For such nodes, the *SOFT* parameters of the local exchange classes are augmented by a weight associated with the rule.

d) Equipment Rules

The equipment rules presuppose that certain equipment types are more likely to be associated with certain classes of switch. Several equipment types can augment soft parameters. Three of these equipment types indicate the node is most

likely to be a transit exchange, and when they apply, a weighting factor is added to the transit class SOFT parameters for the affected node. Two additional equipment types are associated with local exchange classes, and these rules add a weighting factor to $SOFT_{5i}$ and $SOFT_{6i}$.

These heuristics vary in the perceived quality of their diagnostic value. In the CLIPS node classification routine, the CLLI rule is considered the strongest indicator of a node's class, followed by the NPACOC and OCN rules. The various equipment rules are considered the weakest of the soft inference rules. In testing the efficacy of the soft inference implementation, this thesis will evaluate the impact of introducing soft inferences on solution times, and the ability of soft inferences to influence the solution.

IV. PERFORMANCE OF THE BASELINE MODEL

A series of preliminary tests are run using the node classifier IP to classify a number of test networks. This initial testing determined the best solver options and identified performance characteristics of the baseline model. This chapter outlines the equipment, software and methodology used to test the accuracy and solution speed of formulation variants, and conclusions of the preliminary tests. Descriptions of the network (logical) topologies used in the testing are also provided.

A. TEST NETWORKS

Twenty-three test networks are used to evaluate the effectiveness and accuracy of the basic formulation, hard and soft inference processing, and various schemes for accelerating solution times. Collectively, these test networks are hoped to encompass the range of characteristics that may be encountered when GCAT is fielded. Appendix A contains a table summarizing the principle characteristics of these networks, as well as figures depicting some of the networks.

1. U.S. Regional PSTNs

Several U.S. regional PSTN physical network structures were acquired from open sources for testing the IP. For eight test networks (networks 1-6, and "Tracy" and "Balt"), the entire logical topologies are estimated from these existing PSTNs, some of them different logical derivations of the same physical network. Network 0 is built up from actual U.S. switching stations, but the logical structure is notional. This network was designed to provide a simple, tree-like network during the early stages of the IPs' development. It is also the only network derived from an actual PSTN that uses four levels of switches.

These networks range from leafy trees (network 0, Tracy and Balt) with only one triplet ring (completely connected node trios), through more complicated networks containing multiple mesh configurations and rings (nets 4, 5, 6). This range of sizes and configurations presumably constitutes a diverse sample of the actual PSTN population. Diagrams of these networks can be found in Appendix A.

Accompanying each test network derived from actual U.S. regional PSTNs is open-source data from which soft inferences may be derived, and known real-world node classifications. These networks are intended to test the accuracy of the formulation, and evaluate the model's behavior under the influence of soft parameters.

2. Large Notional Networks

To better estimate the effect of model enhancements for speeding up solution time, large networks are needed. When solving smaller networks, it is difficult to assess whether differences in solution times result from normal variance or from a specific change in the model. Larger networks, with longer average solution times, accentuate the affect of changes to the model.

The large networks used for testing in this thesis are simply aggregations of copies of the U.S. regional networks. The aggregations are formed by adding the links needed to interconnect the top-level nodes of the component networks. The largest of these networks is aggregated from four copies each of networks 5 and 6, and may be considered an extreme upper bound on the PSTN classification problem. While symmetric, it is also quite complex, with 212 nodes involved in various mesh configurations.

3. Networks with Modified Longest Shortest Paths

The non-notional PSTN logical topologies available for this thesis contain only three levels. In order to evaluate performance of the formulation with networks of four levels, networks 4, 5, and 6 are modified by appending or removing nodes on their longest shortest paths. The longest shortest paths of a network refer to those shortest paths that are among the longest in the network. These networks so modified are denoted "Lop" (for "Lop-sided") plus the network number and an additional suffix letter. Networks lacking the suffix have had a longest shortest path shortened, e.g., 'Lop6.' The suffix 'a' indicates paths have been extended by the addition of one node; a 'b' indicates paths have been extended by two nodes.

Figure 5 depicts network Lop4a. Extending the longest shortest paths in this network results in the addition of a hierarchical level (compared with network 4—see Figure 17 in Appendix A). These networks are useful in evaluating the performance of routines that calculate an upper bound on *zclass* from the network topology. Analysis of a network's longest shortest paths in preprocessing routines is a topic in a later chapter.

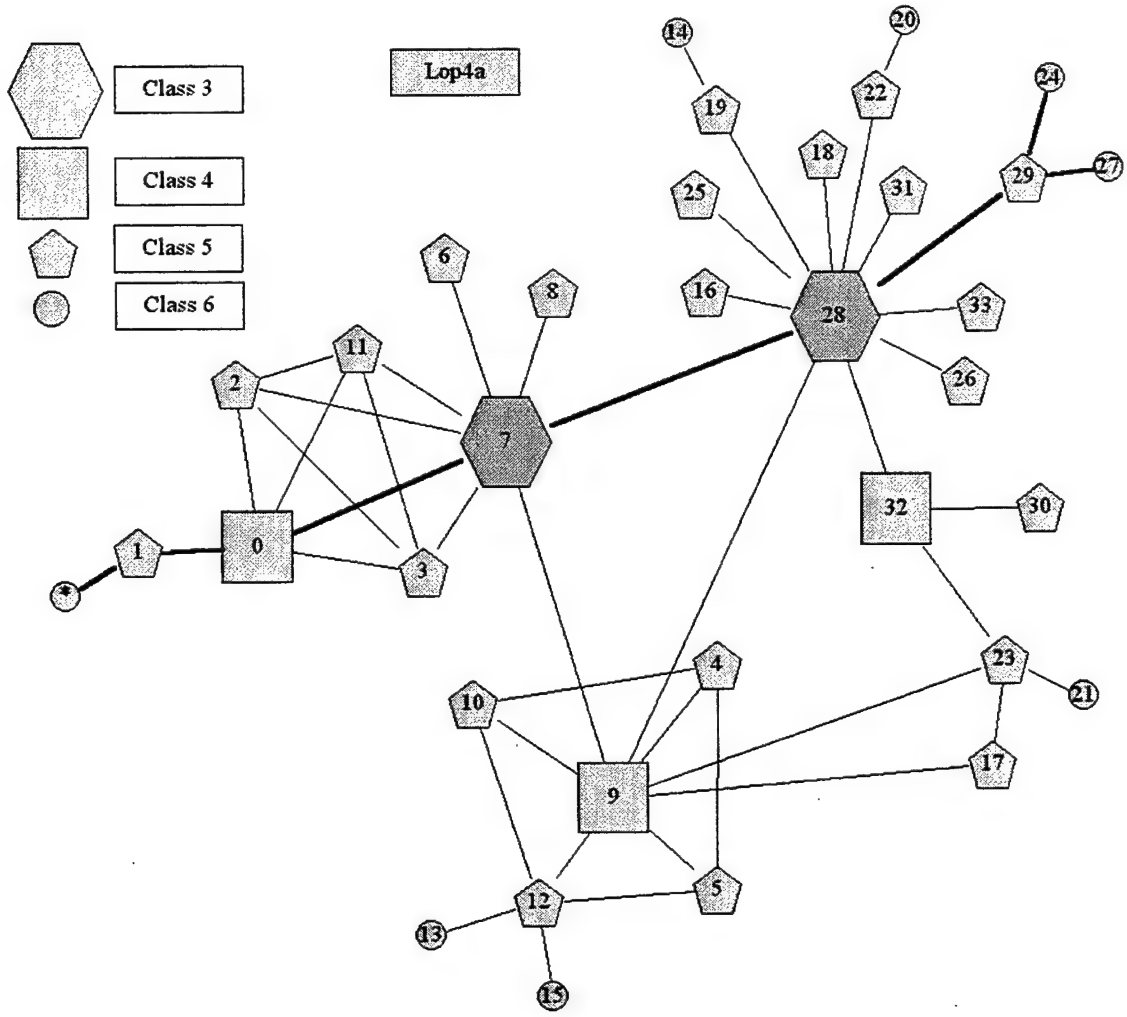

Figure 5. Example of a Network with a Modified Longest Shortest Path

Network Lop4a is formed by adding the node marked with an asterisk to one of Network 4's longest shortest paths. Extending this path requires that an additional hierarchical level be added to the network (see Figure 17 in Appendix A to compare with the structure of Network 4). The longest shortest paths are indicated by darker links. Analysis of a network's longest shortest paths is the subject of a later section.

B. TESTING METHODOLOGY

The IP is implemented in GAMS Development Corporation's Generic Algebraic Modeling System (GAMS), and solved using IBM's Optimization Subroutine Library (OSL) solver. The user-selectable options of GAMS and OSL are described in the GAMS Language Guide (1997). The primary test equipment used is a

166 MHz Pentium Personal Computer (PC) running under Windows 95. This PC is representative of the processing power of low-end work stations. An additional rationale for conducting tests on a lower-end processor is to emphasize differences in solution times between various model options. For certain very lengthy test runs, a 400 MHz PC, running under Windows 95, is employed. At times during the testing, considerable variance was observed in solution times between runs of identical models. Because of the time-consuming nature of many test trials, most of the solution times presented in this thesis represent the results of a single trial. Whenever possible, a verification trial was conducted, and any large inconsistencies in solution times resolved with additional trials.

When evaluating the effect of changes to the model, a baseline formulation is presumed, and changes to this baseline are specified. The baseline model is the formulation of Chapter III. Unless otherwise specified, no hard or soft inferences, or preprocessing of any kind is used when solving the model. The preliminary testing of this chapter establishes the most effective solver options, branching strategy, and objective function parameter weight values; these then remain constant throughout the evaluation of preprocessing routines and soft inference testing of later chapters.

For testing, a cut-off time of 600 seconds is enforced. This ten-minute limit is arbitrarily determined to be twice as long as the maximum tolerable solution time; i.e., if the optimal solution cannot be returned in five minutes, the adequacy of the formulation for use in an interactive application is questionable.

The preliminary testing of this chapter requires the introduction of no additional data other than the logical structure of the network under study. For later testing, the generation of data parameters needed by preprocessing routines is assumed to occur prior to invoking GAMS. These data are generated by a separate JAVA program (modified from J. Brandeau, 1998) and stored in a file. The data files contain

all derived data parameters described in later chapters. Solution times reported in this thesis do not include parameter generation times, nor the model generation time, which incorporates the time needed to read the data files. For most test networks, these times are insignificant. Since in implementation very few of the data parameters need actually be inputted to the model, the solution times obtained for this thesis are probably consistent with those an analyst would observe with similar equipment in a streamlined implementation.

C. PRELIMINARY TESTING

Preliminary testing determines the most effective solver options and branching strategies for reducing solution times. This section describes the selection of model parameter weights, and GAMS and solver options. These settings remain constant in subsequent testing of subsequent chapters. This testing also provides insight to the node-classifier IP's baseline performance, which is also described here.

1. Objective Function Weights

The overriding performance criterion for the IP is that it must return correct switching station classifications. Solution speed is a secondary, although important, consideration. The model parameters *ZWT*, *TWT*, and *PWT* define the characteristics of the network sought, and hence determine the accuracy of the solution.

a) ZWT/TWT/PWT proportions for accuracy

By choice of *TWT* and *PWT*, one determines how many nodes must aspire to become class 5 switches to overrule the assumption of fewest possible top-level nodes. The relative proportions of *ZWT* and *TWT* also define how many tops must aspire to non-top status before an additional level will be allowed in the network.

All of the test networks derived from actual PSTNs (which have known real-world node classifications) are formed with the fewest possible hierarchical levels. Consequently, assessing the best *ZWT/TWT* proportion is more a matter of possible impact on solution speed than accuracy. As long as *ZWT* is large enough relative to *TWT* that no possible number of nodes aspiring to be tops may overrule the *Min_Level*s assumption, accuracy in terms of number of hierarchical levels is assured for the test networks derived from U.S. PSTNs.

In the general case, the IP can be configured to seek network structures not necessarily adhering to the *Min_Level*s assumption by appropriate selection of *ZWT*, *TWT*, and *PWT*. Suppose the rule for a certain group of PSTNs is that an additional hierarchical level is preferred to having four tops, but not to three. In this case, the *ZWT/TWT* proportion would be between three and four. Selecting $ZWT = 1.25$, $TWT = 0.5$, and $PWT = 0.09$ configures the model to seek an additional level in order to avoid establishing a fourth top-level node, and top-level nodes would be preferred if they enable six or more aspiring class 5 node to realize their aspiration. Establishing values for these parameters that are not multiples of each other reduces the possible dilemma of multiple optimal solutions. Which of the top-level candidates will be elevated to a higher level depends on a somewhat complicated function of the numbers of nodes whose status would change if the level of a given node is elevated. Establishing appropriate parameter proportions for classification of networks of greater than four levels would require additional shaping assumptions, and perhaps establishing PWT_c , i.e., weighting classes other than class 5s, in order to define the desired shape. Other than noting the IP could be modified to seek out topologies of more levels than required by the parentage assumptions of a hierarchy, no specific structures of such networks will be hypothesized. The *ZWT/TWT* proportion used in the speed trials of later chapters will be determined assuming a network is constructed using the minimum needed levels.

Given the minimum number of hierarchical levels, each test network has a set of nodes that must be tops because all their descendant nodes must have class greater than or equal to the lowest hierarchical level. For most networks, there is a set of nodes aspiring to become top-level nodes, but not required to be at the top level by depth of their descendants. Whether these nodes become tops in the solution depends on the *TWT/PWT* ratio. An aspiring node will become a top if the number of nodes that would become class 5, less the number of nodes already class 5 that would change class, is greater than or equal to *TWT/PWT*. Some of the actual PSTN test networks have no additional nodes that aspire to become tops, because of the requirement for complete connectivity between tops. For the networks with additional choices, Figure 6 identifies the range for the *TWT/PWT* ratio within which a correct solution for each network will be found (in terms of correct top-level assignments). The Balt network has a non-mandatory top-level node with only one descendant. For this network to solve correctly, the formulation must either reward for additional tops, or soft inferences must correctly influence the solution. From Figure 6, it can be seen that the *TWT/PWT* ratio needed to provide accurate solutions in the test networks is between 2 and 3.

b) Effect of varying parameter weights on solution speed

The introduction of soft inferences into the model, in effect, varies the values of objective function parameter weights. Consequently, it is important that the model exhibit robust performance through a wide range of parameter values. Figures 7 and 8 chart solution speeds versus various values of *ZWT* and *TWT*. Solution speeds for the test networks derived from actual PSTNs are relatively insensitive to the value of *ZWT* and *TWT*, although networks 4, 5 and 6 (containing mesh-connected portions), and Tracy, solve slowly at some parameter values. The larger notional networks show greater variability in solution times as the parameter values vary.

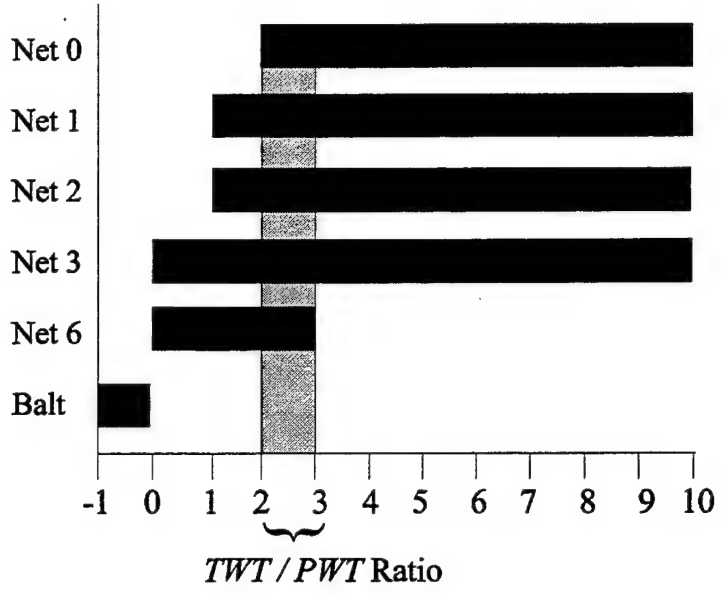

Figure 6. Range of the *TWT/PWT* Ratio Giving Correct Solutions

The *TWT/PWT* ratio establishes the point at which the *More_5s* assumption will overrule the *Min_Tops* assumption. In the test networks derived from actual PSTNs, the most accurate top-level assignments are found when this ratio is between two and three. Networks 4 and 5 are not included in the figure because they have no eligible top-level candidates (not already required to be tops by depth of their unique descendents) meeting the *Top_Mesh* requirement. They are therefore insensitive to the values of *TWT* and *PWT*. A *TWT/PWT* ratio between 2 and 3 provides the most accurate top-level assignments in the test networks.

Figure 7, and Table 9 in Appendix B, show that for *TWT* and *PWT* fixed at 0.5 and 0.2 respectively, lower values of *ZWT* provide the overall best solution times. For example, at *ZWT* = 4, 14 of the networks are solved with solution times within 20% of the best time attained for any value of the parameter. However, these low values of *ZWT* are too small to enforce the *Min_Levels* assumption in the aggregated networks. For *ZWT* large, the selected choice is *ZWT* = 60. Also from Table 9, the speediest choice of *TWT* is also in the range providing accurate solutions with the *PWT* value used; hence, the values used in subsequent testing are *TWT* = 0.5, and *PWT* = 0.2.

2. GAMS/OSL Settings and Branching Priorities

During preliminary testing, the solver options selecting how OSL conducts branch-and-bound preprocessing (*bbpreproc*) are varied. Also varied are options for selecting variables for branching (*strategy*), and performing model reduction prior to starting the optimization procedure (*presolve*). Twenty different combinations of these settings are evaluated, using the parameter weights determined in the previous section, and with all other OSL options remaining at their default values. Preliminary testing also includes evaluating the effect of specifying a branching priority. Branching priorities specify for the solver the relative order in which variables should be selected for branching. Nine different branching priorities were evaluated. For this phase of the testing, a derived integer variable (*sumtops*), equal to the number of top nodes, is added to the formulation. This variable was ultimately found not helpful, and is not present in the model during later testing.

From this empirical testing, the best settings and branching priorities are selected. These choices remain constant throughout the subsequent tests of later chapters. The complete results of this testing are contained in Tables 10 and 11 of Appendix B. In summary, the selected combination of solver settings prompt OSL to use regular branch and bound during preprocessing (*bbpreproc* = 2), heuristically compute pseudo-costs during simplex branching (*strategy* = 8), and perform model reduction only by removing redundant rows (*presolve* = 0, the OSL default). Other *presolve* options provide results on a par with *presolve* = 0 (see Table 5 in Appendix B); however, these more elaborate model reduction schemes can occasionally fail (GAMS Language Guide, 1997). The best branching priority assigned *zclass* a high branching priority, and all other variables the same low branching priority. These priorities solved 15 (of 23) networks with solution times within 10% of the best attained (see Table 11 in Appendix B). Branching first on the variables with the greatest impact on the objective function value is a common approach (Winston,

1993). Given *zclass*' considerable impact on the objective function value with *ZWT* overwhelmingly large, the superiority of priority branching on *zclass* is not surprising.

The effect on solution time of varying settings other than branching priorities is subtle. The outcomes of most trials are inconclusive—improvements in solution times for certain networks are offset by worsened times for others. While certain solver options and branching schemes seem more universally helpful than others, the effect of a good selection is not sufficient to reduce solution times to acceptable levels. However, a poor selection of settings can dramatically worsen solution times.

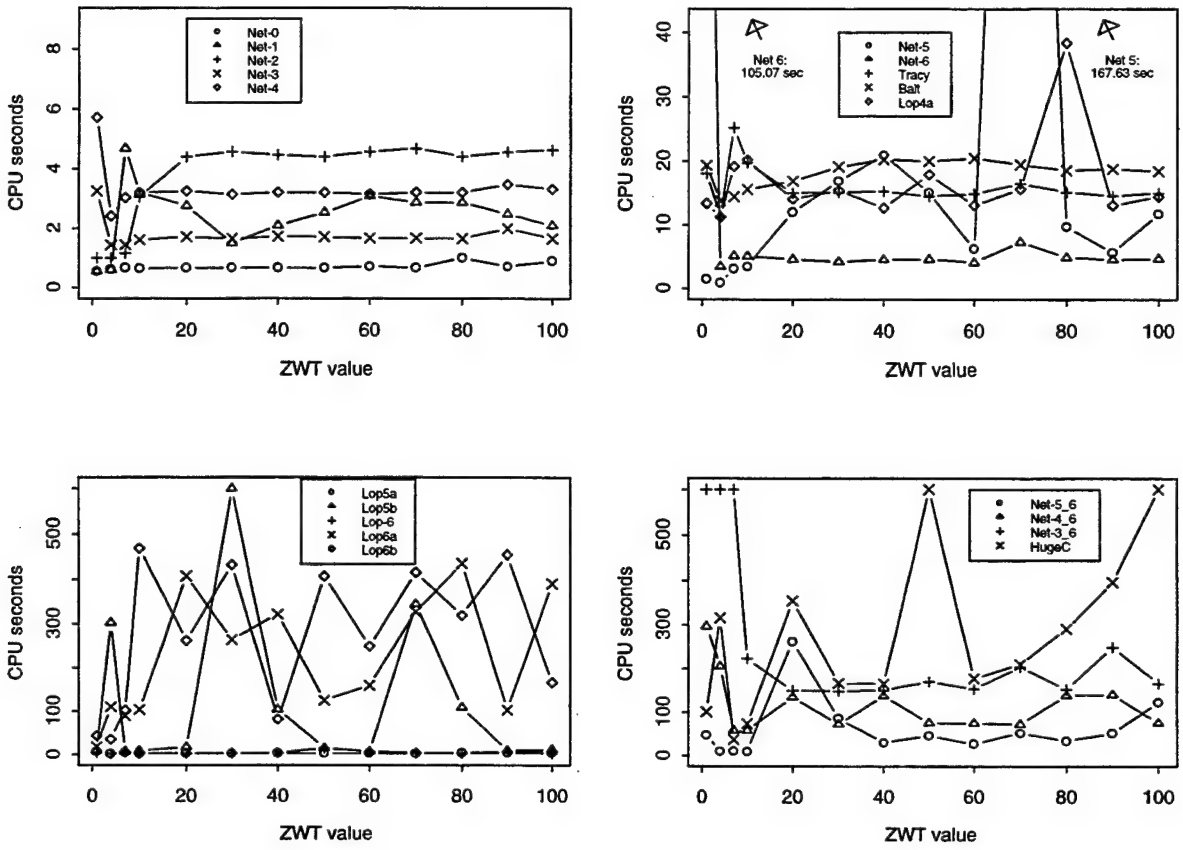

Figure 7. Effect on Solution Time as *ZWT* Varies

For these trials, *TWT* and *PWT* are constant at .5 and .2, respectively. Values at 600 seconds indicate no optimal solution was attained. Test networks that could not be solved within 600 seconds at any value of the parameter are omitted from the plots. A 400 MHz PC was used to collect this data. At large values of *ZWT* (relative to *PWT* and *TWT*), solution times for the networks derived from actual PSTNs are relatively stable. *ZWT* = 4 provides the most solution times within 20% of the best attained for each network. However, this value of *ZWT* returns some of the worst times recorded for the lopsided networks, and also is insufficiently large to prevent the larger aggregated networks from adding a hierarchical level. Table 9 in Appendix B contains the data depicted in this figure.

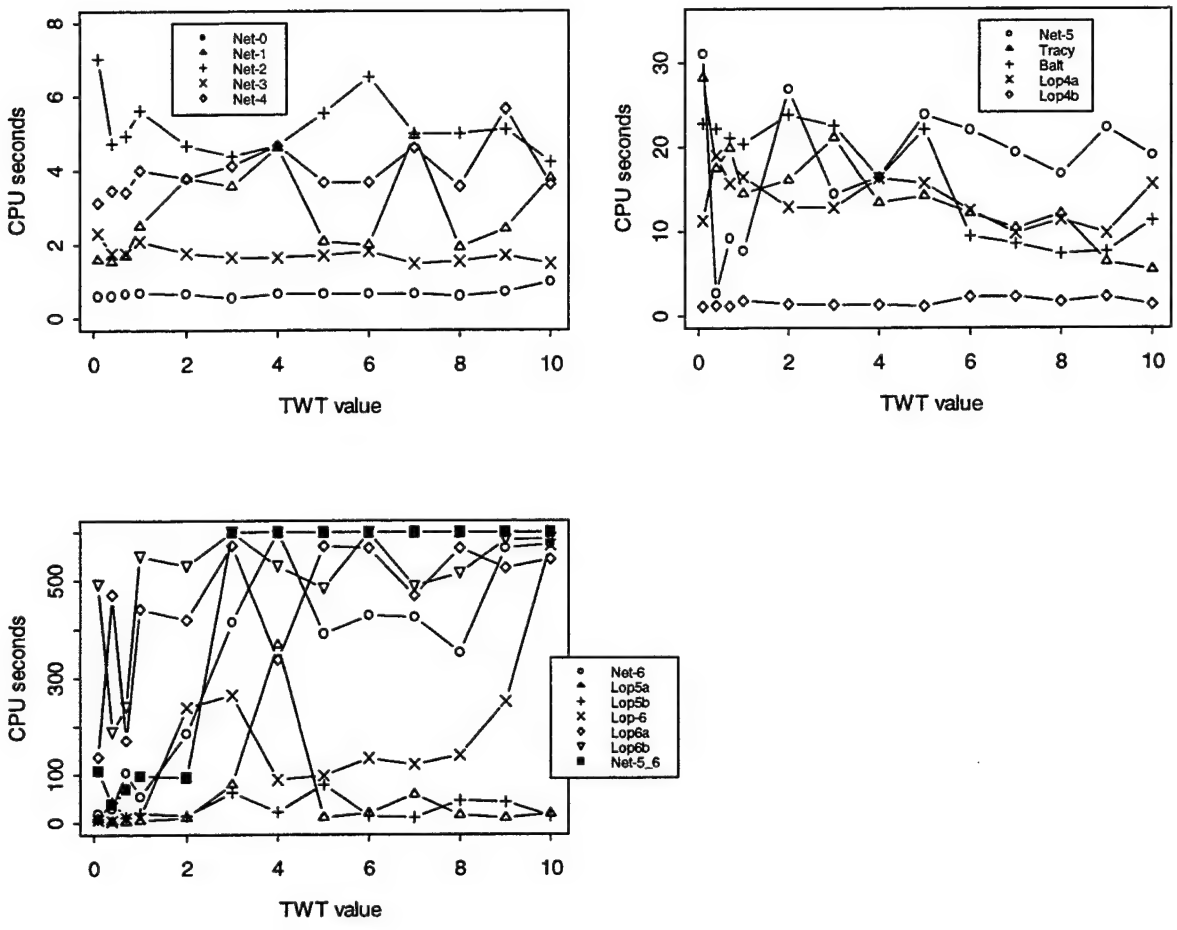

Figure 8. Effect on Solution Time as *TWT* Varies

For these trials, *ZWT* and *PWT* are constant at 100 and .2, respectively. Values at 600 seconds indicate no optimal solution is attained. Test networks that are not be solved within 600 seconds at any value of the parameter are omitted from the plots. A 400 MHz PC is used to collect this data. Overall speediest solution times are attained at low values of *TWT*, also in the range providing most accurate top-level assignments for the selected value of *PWT*. Data displayed in this figure is in Table 9 of Appendix B.

D. CONCLUSIONS FROM THE PRELIMINARY TESTS

Complete results of the pilot study are in Appendix B. The pilot study does not evaluate all possible combinations of solver settings, objective function parameter values, or branching priorities. But, from the sampling done, a number of conclusions can be drawn.

The most important initial observation is that the formulation can accurately classify nodes for the U.S. regional PSTNs. With $ZWT = 100$, $TWT = .5$, and $PWT = .2$, all the networks derived from actual PSTNs, excepting Balt, solve with correct top-level node assignments and number of hierarchical levels. Balt's ground-truth structure violates the *Min_Tops* assumption that the fewest possible tops will be used to construct the network. Networks 4 and 6 have class 6 leaf nodes connected directly to class 4 tandems. Nodes in this configuration violate the *More_5s* assumption and are incorrectly classified as class 5 end offices. Both these types of errors point out that the assumptions of the model are not universally true, at least for U.S. PSTNs. Errors caused by class-skipping nodes are of little concern, since the class assigned a leaf node has no impact on route tables. Misclassifications at the top level of the network can cause errors in the route tables, and ultimately, the route predictions generated by later GCAT methods.

The second most important observation is that the solution speed of the unsophisticated baseline model is only marginally acceptable for inclusion in GCAT. Figure 9 shows the results of ten of the trial runs, in this case devoted to determining the effect of solver options on solution times. Observe in Figure 9 that some test networks could not be solved in 600 seconds of processing, regardless of choice of solver settings. While five minutes is arbitrarily chosen as the upper limit on acceptable processing time, a much faster solution is preferred. Solution times are also unpredictable as model attributes vary; even networks that typically solve quickly occasionally require excessive processing time with some choices of solver options.

The performance of the node-classifier IP depicted in Figure 9 is typical of that observed throughout the preliminary testing. The GAMS model will need to incorporate tactics to speed solution time if it is to be acceptable as a method in GCAT.

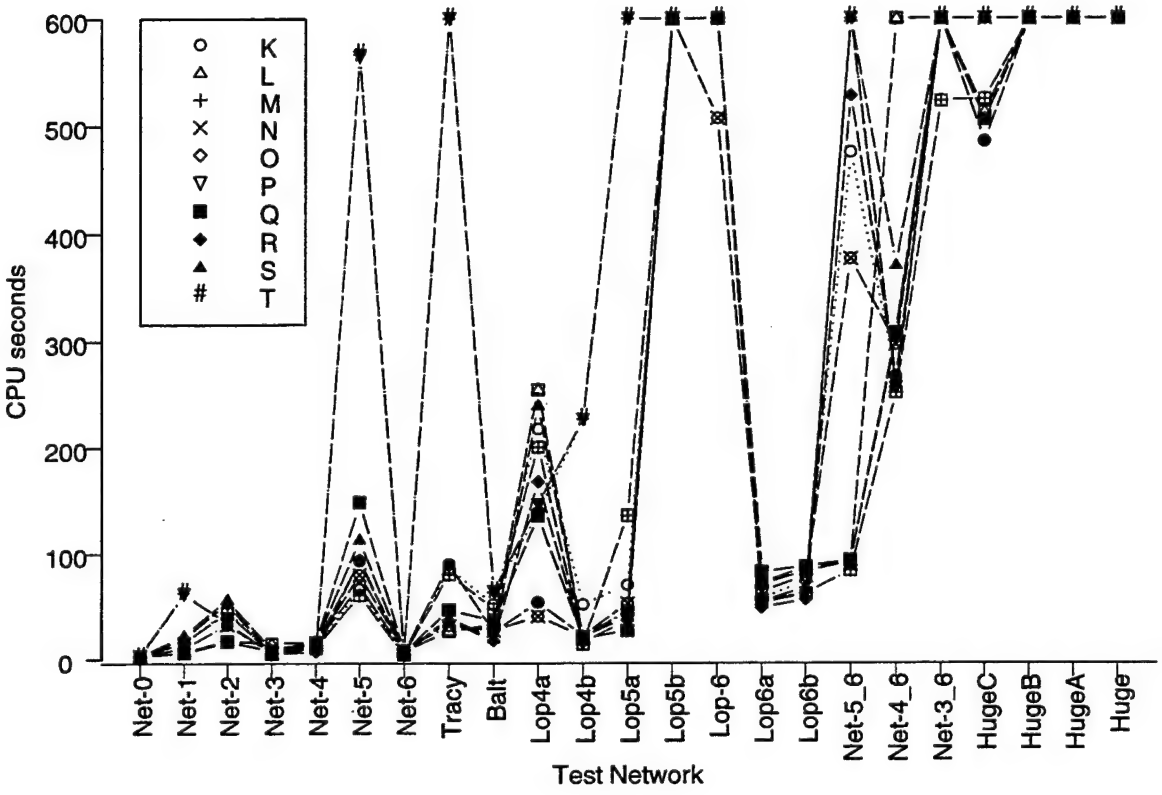

Figure 9. Solution Speeds Attained During Selected Trials

The solution speeds attained with various combinations of OSL solver options are depicted in this chart. Notice that several networks are not solved in ten minutes of processing on the 166 MHz PC, regardless of choice of solver options. The solution times for the baseline model are relatively insensitive to choice of solver options, although certain options perform very poorly for some networks. This is consistent with the behavior observed during other phases of the preliminary testing. Identification of the specific solver options used in the series presented in this chart, and during all the trial runs of the preliminary testing, is available in Appendix B.

V. PREPROCESSING

This chapter describes methods for reducing solution time through preprocessing of the input data. Preprocessing refers to those operations accomplished to improve a formulation by fixing or tightening bounds on variables, reducing or simplifying equations, and similar tactics (e.g. Nemhauser and Wolsey, 1988). Because GCAT is intended to be an interactive application, solution times of the baseline formulation of Chapter II are inadequate. The baseline formulation has, through extensive experimentation, been constrained to the extent found helpful in reducing solution times. Still, for many problems the branch and bound process is quite lengthy. This chapter evaluates several routines that analyze the input node-node adjacency matrix representing the logical structure of the network, and use insights gained to fix or eliminate variables or equations, or emphasize critical features for the solver. While these routines are quite effective in reducing solution times, they require making additional assumptions about the network. These additional assumptions may restrict the ability of soft inferences to influence the solution. In the interest of later brevity, each proposed preprocessing routine is assigned a shorthand name.

A. LEAF PLUCKING

The simplest of these routines considers nodes of degree one, i.e., nodes with one emanating arc. Some conclusions about any such node i can be immediately drawn:

$$\begin{aligned} top_i &= 0 \quad \forall i: \text{degree of } i = 1 \\ p_{ij} &= 1 \quad \forall i: \text{degree of } i = 1, j: (i, j) \in A \end{aligned}$$

A leaf node is not a top-level node of any non-trivial network. Also, we can safely assume the node adjacent to a leaf is parent to the leaf node. This preprocessing routine is termed *Leaf_Pluck* for short.

From the earlier description of switching station functional types, it is also apparent that a leaf node must be a local office, rather than a transit office. With only one trunk, the node must be an interface for subscriber loops. Leaf nodes therefore could safely be restricted in the model to be local exchanges. From the perspective of route table generation (the ultimate goal), the actual class assigned a leaf is of little consequence since there is only one route out of the node. Restricting classes assigned leaf nodes would be of marginal utility in terms of reducing solution time. A more powerful assumption is the converse: restrict assignments to the lowest hierarchical level to leaf nodes, as described next.

B. RESTRICTING CLASSIFICATIONS AT THE LOWEST HIERARCHICAL LEVEL

The number of customers able to connect to an end office is limited by the switch's capacity for connecting subscriber loops. When populations form discrete enclaves, as in small rural communities, it often makes sense to concentrate the traffic of the enclave to preserve resources at the main facility. Many subscribers may have dedicated loops at a concentrator, and be serviced using far fewer switches at the end office, with no significant degradation of service quality. This makes more efficient use of the limited allocations for subscriber loops at the main switch of the end office serving the area (Freeman, 1989).

In GCAT's model of U.S. PSTNs, these satellites are referred to as "class 6" nodes. In a sense, they are merely extensions of a parent facility. By assuming remote concentrators do not provide a trunk (or non-local) routing function, but only concentrate traffic for the parent, the model can be further constrained by the requirement that only leaf nodes may be classified at the lowest hierarchical level, i.e.,

be class 6 nodes, in the GCAT nomenclature. This assumption essentially removes an entire hierarchical level from the network, a significant assist to the solver. The shorthand name for this routine is *Class_6*:

$$bcl_{ci} \in \begin{cases} \{0\} & \text{if } c = 6 \text{ and } i : \text{degree of } i > 1 \\ \{0,1\} & \text{if } c = 6 \text{ and } i : \text{degree of } i = 1 \end{cases}$$

All eight test networks derived from U.S. regional PSTNs follow this rule. The only test network violating this assumption (i.e., having class 6 nodes with degree > 1) is network 0, which is notional.

C. LONGEST SHORTEST PATHS ANALYSIS

The Longest Shortest Paths (L-S Paths, for brevity) of a network refer to those paths of minimum length (number of links) between any pair of nodes which are among the longest in the network. The L-S Paths can be used to establish an upper bound on *zclass*, and heuristically can give strong indications to the identity of the top-level nodes.

1. Establishing a lower bound on the number of hierarchical levels

The length of the L-S Paths imposes an upper bound on *zclass*, since the lengths of these paths determine the minimum number of levels required to form a hierarchical network. At a minimum, one hierarchical level is required for every two trunks in the L-S Paths, as shown in Figure 10. Therefore, subtracting a proportion of the number of trunks in the L-S Paths from the maximum class used in the network establishes the maximum value *zclass* may attain:

$$zclass \leq \max(c) - \left\lfloor \frac{\text{length(L-S Path)}}{2} \right\rfloor$$

Figure 10 displays all possible configurations for L-S Paths in the ATT scheme. The expression above establishes an upper bound on *zclass* (i.e., the minimum possible levels) in every case.

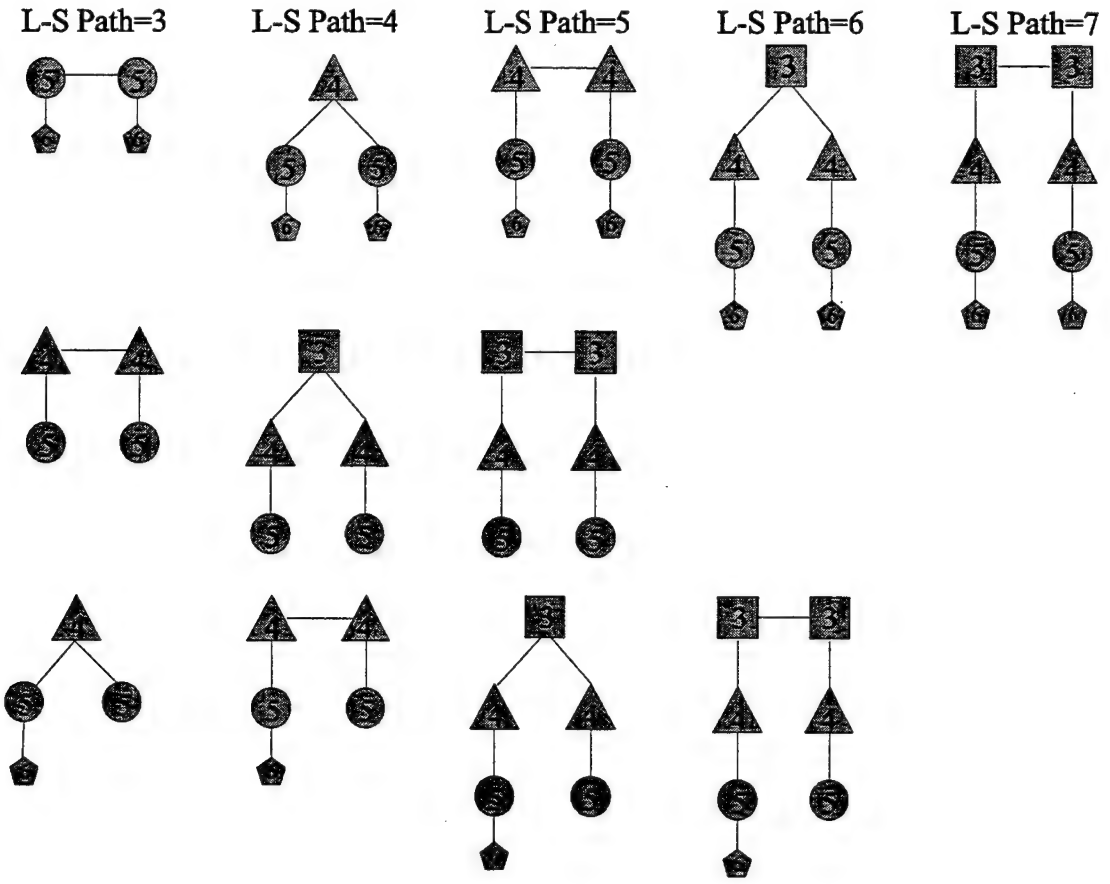

Figure 10. Possible Configurations of L-S Paths

The L-S Paths of a network establish a lower bound on the number of levels required to form a hierarchy. Notice that top-level nodes must be in central positions in the L-S Path, while nodes at either end of the path cannot be tops. Figure 5 shows that extending a L-S Path can require adding a hierarchical level to the network.

2. Fixing the number of hierarchical levels in a loop

Since an upper bound for *zclass* is easily established from the input data, a strategy for reducing the scope of the problem presented to the solver is to fix *zclass* at its upper bound and make consecutive calls to the solver, decrementing *zclass* with each subsequent call. If the *Min_Level*s assumption is strongly enforced (i.e. *ZWT* is large enough relative to *TWT* that under no circumstances would an unnecessary level be added to the network), the loop may be exited as soon as an optimal solution is found. With *ZWT* large, there can be no better solution found by adding a hierarchical level. For purposes of automating the formulation, the solver loop may be exited upon attaining an optimal solution if $ZWT/TWT > 0.1 \cdot |N|$. This quite conservative rule allows the possibility of finding a better solution with more than the required number of levels if the magnitude of *TWT* with ten percent of the networks' nodes being tops is sufficient to overrule the *Min_Level*s assumption. The pseudocode for the routine we shall term *Z_Loop* is:

$$zclass' = \underline{zclass} + \Delta - \left\lfloor \frac{\text{length(L} - \text{S Path)}}{2} \right\rfloor$$

bestSoln value $= -\infty$
while($zclass' \geq \underline{zclass}$ {
    solve MIP with $zclass \equiv zclass'$
    bestSoln value $= \max(bestSoln, currentSoln)$
    if (*Feasible Solution AND* $\left(ZWT/_{TWT}\right) > 0.1 \cdot |N|$){
        $zclass = -\infty$ (*exit*)
    }
    else{
        $zclass' = zclass' - 1$
    }
}
display bestSoln

The only additional assumption of this routine is implicit in the condition for exiting the while statement (the network will be constructed using the fewest possible levels), which can be made as conservative as desired. If the solution with the fewest levels is overwhelmingly preferred, this strategy is likely to find a solution with one relatively quick iteration of the loop. This strategy also can provide alternative solutions using more than the minimum required number of levels.

D. IDENTIFYING TOP-LEVEL NODES

The parameters described below are all functions of the input data; i.e., the node-node adjacency matrix defining the logical network topology. Each parameter has some diagnostic value in predicting whether a node is at the top level of the network:

$$degree_i = \sum_{j:(i,j)\in A} 1 \quad (8)$$

$minhop_{ij}$ = length of the shortest path between nodes i and j

$$totalhop_i = \sum_j minhop_{ij} \quad (9)$$

$$maxhop_i = \max_j (minhop_{ij}) \quad (10)$$

$$centrality_i = \frac{\sum_{j:totalhop_j < totalhop_i} 1}{|N|} \quad (11)$$

(8) A node adjacent to many other nodes is more likely to be a top-level node than one with links to fewer nodes.

(9) The parameter *totalhop* of node i is the sum of the hops necessary to travel from i to each node j, where each j is a terminating node of the path (i.e., intermediate nodes k in a path are not considered visited). Nodes with low *totalhop* are the more central nodes in the network (and hence more likely to be at the top level).

(10) $Maxhop_i$ is the maximum number of trunks needed to form a path from node i to any other node of the network. A node whose maximum

minimum-distance from any other node is small is more likely to be a top-level node than one with a larger maximum minimum-distance.

(11) *Centrality*$_i$ is node *i*'s *totalhop* percentile. The most central node in the network, with lowest *totalhop* parameter, will have a centrality value of 0. The centrality of the most remote nodes will be near 1.

The data parameters *degree*$_i$, *totalhop*$_i$, *maxhop*$_i$, *centrality*$_i$, and positions on the L-S Paths can be used to identify likely top-level nodes. Appendix C shows the values of these parameters for the test networks derived from actual PSTNs. As heuristics, these parameters are quite good at diagnosing top-level status. Because of the requirement for top-level nodes to be completely connected, the identity of one top-level node is a powerful clue to the identity of the other tops, i.e., any node not connected to the identified top cannot be at the top level.

The most powerful heuristic indication of a node's top-level status is the *centrality* parameter. Because top-level nodes must be interconnected, as a group they are the minimum distance from all other nodes in the network. The node *i* with *centrality*$_i = 0$ is the most central node in the network. Nodes *j* with *centrality*$_j$ close to one (i.e., having many nodes more central than they) are unlikely to be at the top level. The *Min_Hop*(α) routine fixes the minimum centrality node to be a top, and fixes nodes i with *centrality*$_i \geq \alpha$ to non-top-level status:

$$top_i = 1 \qquad \forall i : centrality_i = 0$$
$$top_i = 0 \qquad \forall i : centrality_i \geq \alpha$$

The *centrality*$_i$ statistic is a proportion, between zero and one. When the $\alpha \geq 1$ is selected, all nodes will remain eligible to become tops. *Min_Hop(0.1)*, for example, represents a routine in which all nodes *i* with *centrality*$_i$ greater than 0.1 are prevented from being tops. Typically, there will be only one node with a centrality of zero. In the symmetric, aggregated networks of this thesis, there will be several.

E. REDUCING MODEL SIZE

Many of the previously described routines restrict the model by fixing the values of variables prior to solving. The elimination of variables can result in constraints becoming superfluous. These constraints can be removed from the model.

- Constraints (2) become superfluous for any $(i, j) \in A$ for which the *Leaf_Pluck* or *Min_Hop*(α) preprocessing routines have fixed the value of top_i to 0.

- The same preprocessing operations also cause constraints (7) to become superfluous for any i whose top_i variable is fixed to 0.

Notice that constraints (6) are still needed, even with top_i fixed to 0, to prevent i's class from being equal to *zclass*. A single routine, with shorthand name *No_Eqn*, eliminates from the model the superfluous constraints described above. This routine is only applicable in concert with *Leaf_Pluck* or *Min_Hop*(α) preprocessing.

VI. TESTING OF PREPROCESSING ROUTINES

While the additional assumptions mandated by some of the preprocessing routines may be unsuitable for all applications of the IP, the routines are quite effective in reducing solution times. The intent of this chapter is to demonstrate the general effectiveness of the preprocessing routines, under the premise that similar routines could be adopted if necessary for networks with different characteristics.

Data required by the preprocessing routines are generated by a JAVA-language preprocessor adapted from the thesis work of J. Brandeau (1998). Preprocessing time is insignificant for most networks, generally under one second. Processing the larger, aggregated networks can take up to 20 seconds on a 400 MHz PC. While this time is perhaps not negligible, each network need be processed only once. In a production implementation, fewer data parameters would be needed than are generated for this thesis, likely reducing the preprocessing time even further. Consequently, preprocessor time is not considered in evaluating the effectiveness of the preprocessing routines.

The effectiveness of preprocessing routines may appear in two ways. First, the amount of time spent in branch and bound may be diminished. This is the overriding evaluation criterion—preprocessing that appears beneficial in other regards, but extends solution time, is not helpful in the model.

Second, preprocessing may reduce the integrality gap. *Integrality gap* refers to the difference between the IP's objective function value and the objective function value of the linear program obtained by relaxing the IP's integrality requirements. A tighter (smaller) integrality gap is better because the size of the feasible region the

solver must explore is reduced, along with the amount of time spent in branch and bound.

Preprocessing can also improve performance of a model by tightening bounds on variables and eliminating redundant or slack constraints (Nemhauser and Wolsey, 1988). The *Z_Loop* and *No_Eqn* preprocessing routines employ these tactics. The number of equations removed from a model by *No_Eqn* preprocessing may be an indicator of the effectiveness of the routine.

This second test phase analyzes the benefit of preprocessing routines against these metrics. Unless noted otherwise, the solution obtained by the node classifier IP with preprocessing applied is identical to that obtained from the baseline model. This chapter also discusses the accuracy of the solutions found by the IP relative to the actual node classes of the U.S. regional PSTNs. The term *ground truth* refers to the true class of the switching station represented by the node.

A. BOUNDS ON VARIABLES

A factor contributing to the extended solution times of the baseline IP is the size of integrality gap (see Figure 11, and Table 7 in Appendix D). In some cases, the relaxed objective is 40% larger than the optimal integer-constrained objective. Figure 11 charts the reduction in the integrality gap resulting from implementation of some of the proposed preprocessing routines, and Figure 12 shows the associated reduction in solution times.

The most dramatic improvements come from implementing *Min_Hop(1.0)* and *Min_Hop(0.1)*. Identifying a top-level node is a significant assist to the solver—the relaxed objective equals the optimal objective value in eight of the test networks, and all the networks, with the exception of Huge, solve in about a minute or less. By

fixing the value of top_i to 0 for nodes i with $centrality_i > 0.1$, the integrality gap is practically eliminated for all networks (see Figure 11). However, there is no obvious improvement in solution speed attained by identifying non-tops (see Figure 12). Furthermore, fixing top_i variables for nodes i with $centrality_i < 0.1$ (e.g. *MinHop(0.05)*) results in the proliferation of classification errors, since the top_i variables of some ground truth top-level nodes are fixed to 0. For example, in network 6, $centrality_{37} = 0.0714$, and node 37 is a top-level switch, per ground truth. A consequence of applying *Min_Hop(0.07)*, for example, to network 6 is the fixing of $top_{37} = 0$, and an incorrect solution is dictated.

The effectiveness of the *Min_Hop*(α) routine mainly results from fixing $top_i = 1$ for the node(s) i with $centrality_i = 0$. The identity of a top-level node is clearly helpful to the solver. In concert with constraints (2), this single node i with $top_i = 1$ eliminates all non-adjacent nodes from the pool of top-level candidates. However, experiments in which the top_j variables are fixed to 0 for all nodes j not adjacent to the minimum centrality node(s) achieves no significant improvements in model performance.

Variables top_i are also set to 0 by *Min_Hop*(α) when $centrality_i > \alpha$. But, as described above, this variable fixing for "non-tops" has little effect on the model for $\alpha > 0.1$. Table 13 in Appendix D shows no additional reduction in the optimality gap over that attained by *Min_Hop*(1.0), until $\alpha = 0.1$. From Table 14 in Appendix D, the solution times seem unrelated to the value of α.

The *Leaf-Pluck* routine is quite helpful for certain networks (see figures 11 and 12). For the star-configured networks (Balt, Tracy, and Net-3), this routine alone is sufficient to reduce the integrality gap to 0. *Leaf_Pluck* reduces solution times dramatically for some networks, and all but one are solved within 600 seconds.

Class_6 restrictions also reduce the optimality gap and accelerate solution times. With this restriction active in the model, all the test networks can be solved within 600 seconds. The major drawback to the *Class_6* preprocessing is apparent in test network 0, which gains seven classification errors. This network's topology has two connected class 6 concentrators, which violates this preprocessing routine's assumption (see Figure 15 in Appendix A). While the logical topology for this network is notional, the technology probably exists or soon will exist to economically give concentrators a routing function. Consequently, the longevity of this assumption and its applicability outside the U.S. is questionable, and this preprocessing routine may be unsafe for incorporation in the IP.

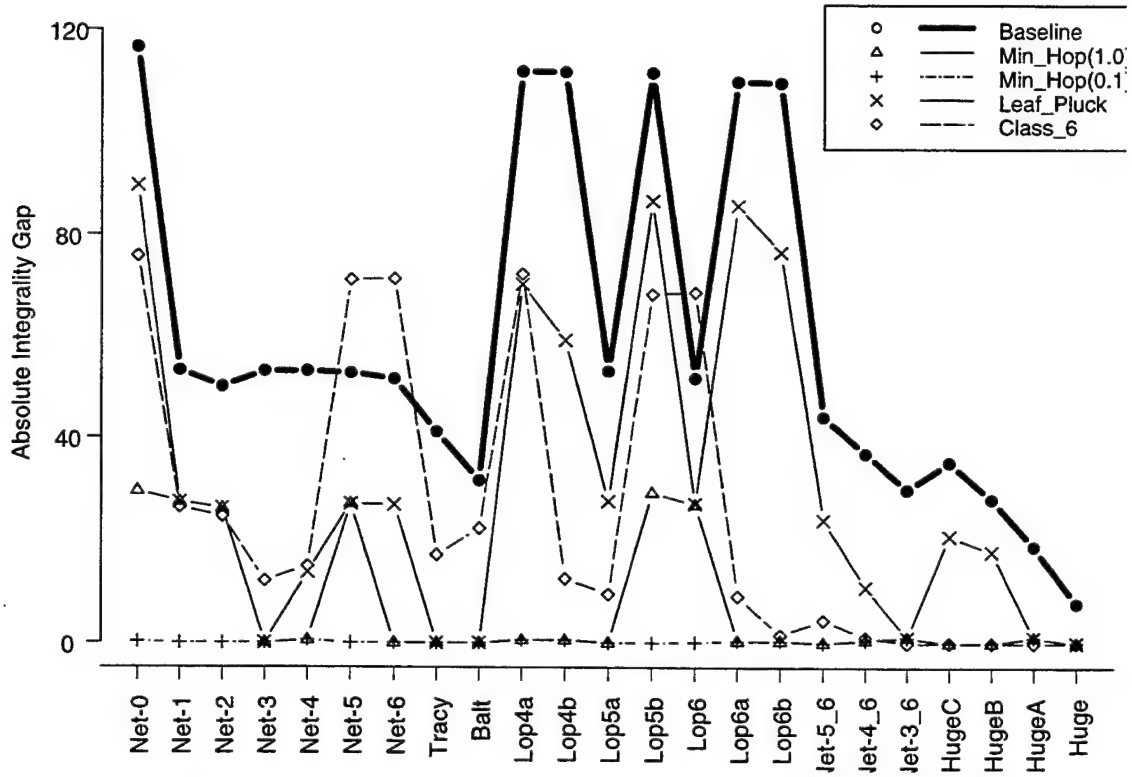

Figure 11. Effect of Preprocessing on the Relaxed Objective Function Value

This chart plots the absolute reduction in the integrality gap resulting from the preprocessing routines. The "Baseline" series represents performance of the baseline model, with no preprocessing. Preprocessing routines are not combined. All the preprocessing routines are effective at reducing the integrality gap over that of the baseline model. The *Min_Hop(0.1)* preprocessing nearly eliminates the gap for all the test networks. Data depicted in this chart is in Table 13 of Appendix D.

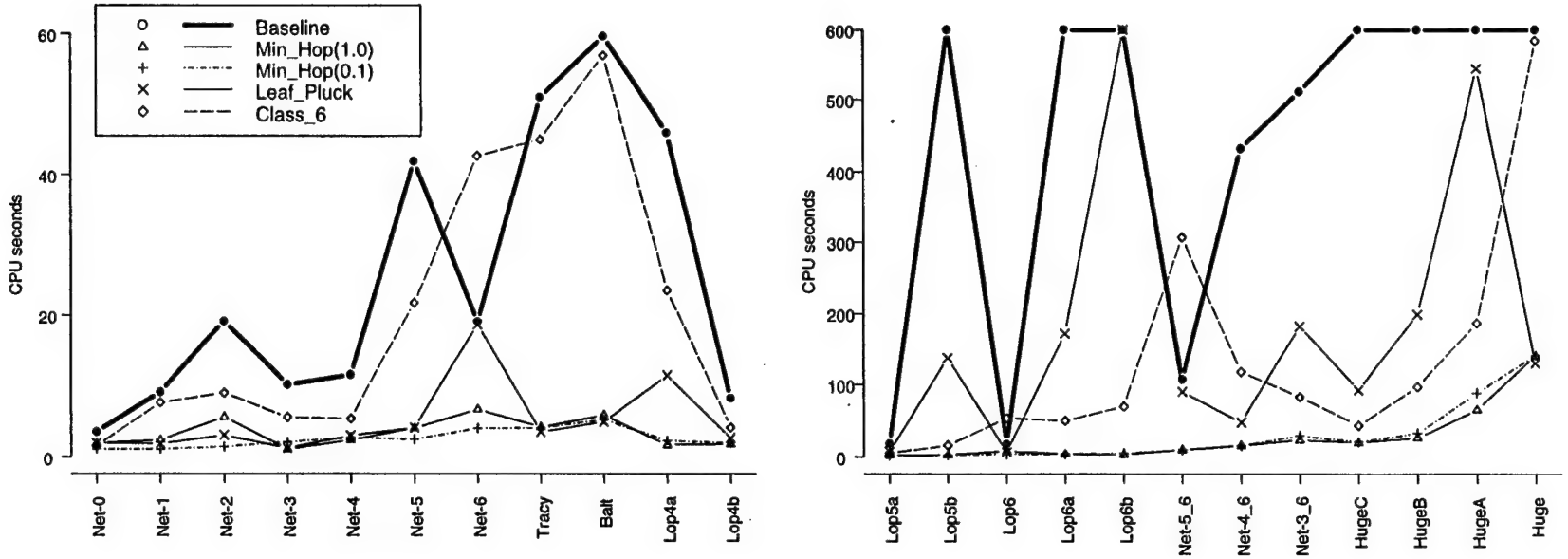

Figure 12. Effect of Preprocessing on Solution Time

These charts plot the solution times obtained by implementing preprocessing routines. Any single routine in almost all cases is sufficient to reduce solution times below the 600-second threshold. Notice that the effect of *Min_Hop*(0.1) is virtually identical to that of *Min_Hop*(1.0). In other words, identifying nodes that are not at the top level provides negligible improvement to solution times over that afforded by identifying one top-level node. Data depicted in this chart is in Table 14 of Appendix D.

B. EQUATION REDUCTION

To see the effect of eliminating superfluous equations from the model, two trial runs are conducted using the *Leaf_Pluck* and *Min_Hop(0.1)* preprocessing routines. In one of the trials, the *No_Eqn* routine is also implemented. Table 2 presents the difference observed between the two trials. The number of equations eliminated by this simple tactic can be quite significant when solving larger networks. No reduction in solution time is apparent until the networks are large (HugeB and larger). However, the preprocessing routines being applied in this trial are quite effective, perhaps leaving little room for improvement in the smaller networks.

Equations are made superfluous by the fixing of variables, so *No_Eqn* is only applicable in the presence of certain other preprocessing. Equation reduction is a model enhancement with no obvious drawbacks, but also with no dramatic benefit in terms of reducing solution times. The efficacy of *No_Eqn* might be more noticeable in concert with less effective preprocessing routines (e.g. *Min_Hop*(0.5)).

C. LOOPING ON ZCLASS

In order to evaluate the characteristics of the *Z_Loop* routine, trials are run using the looping strategy, in concert with various combinations of preprocessing routines. For some of these trials, the range of possible classes is set to be {0, 6}, and solutions sought for all feasible values of *zclass*. Figure 13 shows the percent improvement in solution times and relaxed objective function value attained by the *Z_Loop* routine (with no other preprocessing). The improvements reported in Figure 13 are for the first attained solution.

Network	Reduction in equations	% reduction	Reduction in solution time (seconds)
Net-0	206	44.5	0.10
Net-1	342	47.8	0.17
Net-2	688	55.8	0.11
Net-3	550	54.9	0.23
Net-4	543	50.9	0.44
Net-5	795	59.8	0.67
Net-6	1251	69.6	2.62
Tracy	4263	83.8	0.50
Balt	5650	87.1	2.19
Lop4a	260	23.4	0.01
Lop4b	173	14.9	-1.53
Lop5a	386	27.9	-1.86
Lop5b	421	29.4	-1.32
Lop6	723	41.4	0.77
Lop6a	705	38.1	1.93
Lop6b	579	30.4	1.59
Net-5_6	2726	59.0	1.25
Net-4_6	5832	69.3	5.88
Net-3_6	10072	75.0	5.83
HugeC	6681	67.6	-0.16
HugeB	11191	73.1	8.19
HugeA	23643	80.0	90.01
Huge	45290	82.6	95.28

Table 2. Equation Reduction in the Model

When employed in concert with the *Leaf_Pluck* and *Min_Hop*(0.1) preprocessing routines, the *No_Eqn* routine removes a significant number of superfluous equations from the models. However, *No_Eqn* has no obvious affect on solution time for most of the networks.

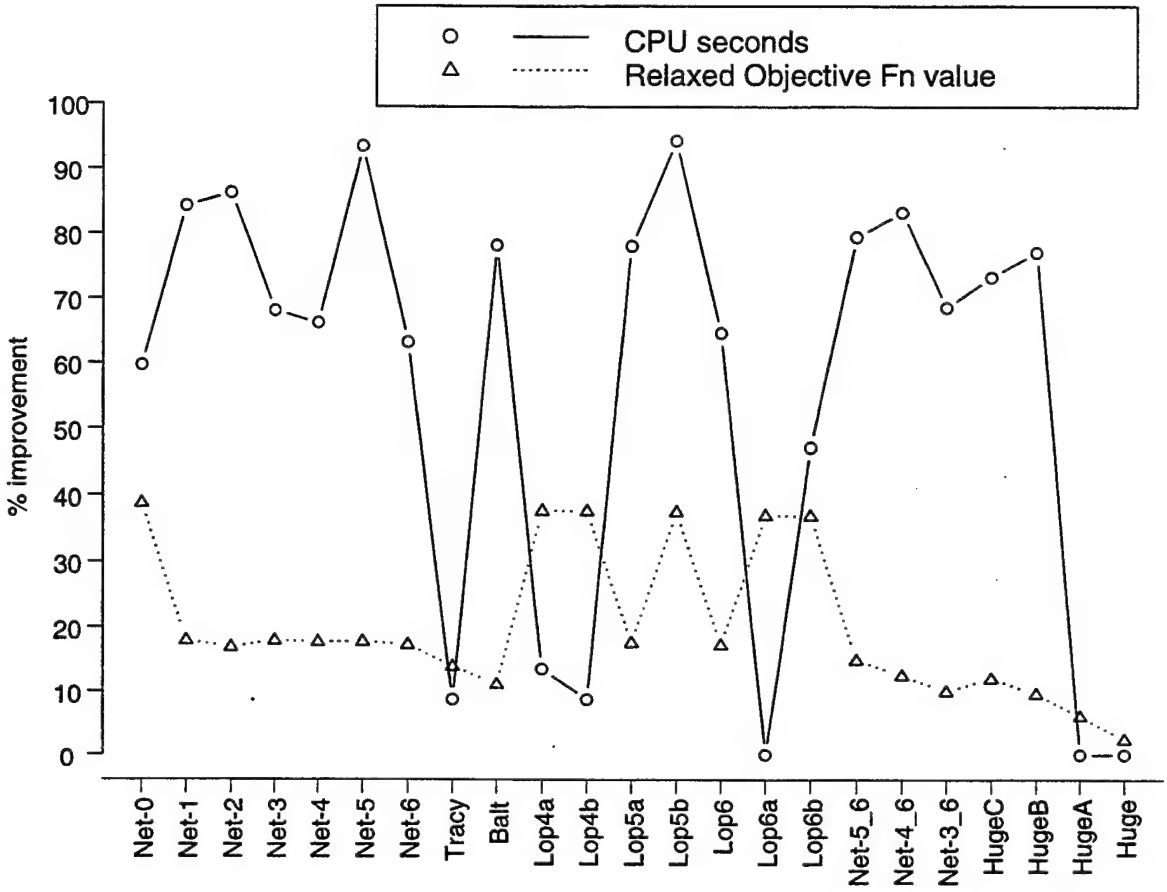

Figure 13. Improvement in Solution Time and Relaxed Objective Function Value

The benefit of fixing *zclass* prior to solving the model is significant when no other preprocessing is applied. Reduction in solution time to the first feasible solution is over 50% for most networks. Solution time of the baseline formulation was taken to be 600 seconds if no solution was attained; therefore, in some cases the percent improvement in solution time may be better than indicated in the chart. Networks HugeA and Huge could not be solved even with the *Z_Loop* routine applied, therefore the solution-time improvement is 0 for these networks. Considerable variance in solution times is observed across several runs of this trial, so improvements of less than 20% are probably not significant. Improvement to the relaxed objective function value is relative to the baseline model's relaxed objective (not the integrality gap). Data depicted is in Table 15 of Appendix D. Test machine is a 166 MHz PC.

Fixing *zclass* at the highest feasible value reduces the size of the feasible region for all of the test networks, as indicated by the reduction in the relaxed objective function value. Reduction in solution time to the first feasible solution is also significant for most networks in the absence of other preprocessing. However,

solution times in this naïve formulation increase exponentially as *zclass* is fixed further from its maximum feasible value (see Table 15 in Appendix D). With the addition of the *Min_Hop(0.1)*, *Leaf_Pluck*, and *No_Eqn* preprocessing routines, solution times became relatively constant, regardless of *zclass*' value. However, the time to first solution for the same preprocessing routines, without using *Z_Loop*, is about the same (see Figure 14).

In conclusion, the *Z_Loop* strategy appears quite helpful in reducing solution times to the first solution in the absence of other preprocessing (see Figure 13 and Table 15 in Appendix D). The exponential increase in solution times with increasing *zclass* suggests additional preprocessing will be needed to explore network structures using more levels than required. When preprocessing routines are applied, however, the solution times for each value of *zclass* become reasonable—most of the test networks can be solved four or five times with varying values of *zclass* in less than 200 seconds.

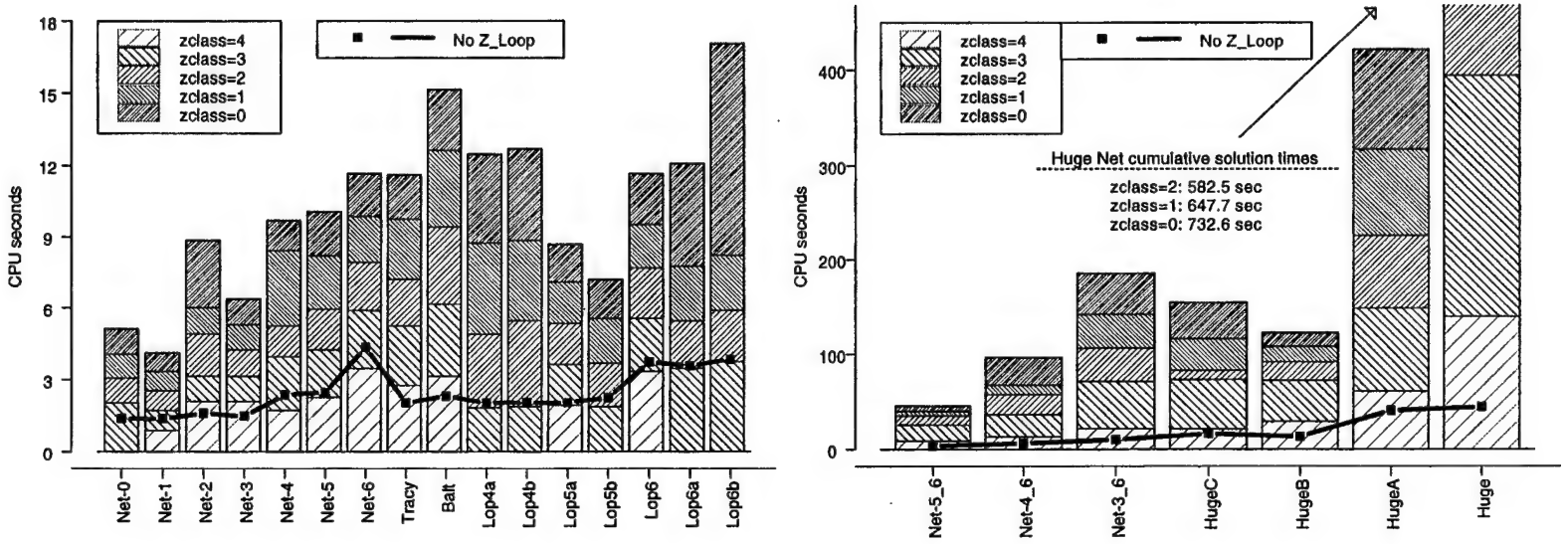

Figure 14. Cumulative solution times for the *Z_Loop* routine

Depicted are the cumulative solution times for each feasible value of *zclass*, using the *Z_Loop* routine plus the *Min_Hop*(0.1), *Leaf_Pluck*, and *No_Eqn* preprocessing routines. Notice that all the networks but Huge can be solved within 600 seconds for all values of *zclass*. All of the test networks derived from actual regional U.S. PSTNs (Net-0 through Balt in the left plot) are solved for every value of *zclass* in under 18 seconds on the 166 MHz PC. Notice also that the *Z_Loop* routine does not improve solution time to the first feasible solution over that obtained by the same preprocessing routines without fixing zclass (the "No *Z_Loop*" line). See Table 16 in Appendix D for the data depicted in these charts.

VII. TESTING SOFT INFERENCES

Soft inferences may only be incorporated into the model when appropriate rules have been devised, and when data applicable to the rules is available. From the standpoint of intelligence collection, it may be very difficult to acquire the needed data. Therefore, it is important to know how effectively soft inferences can influence solutions found by the node-classifier IP. There is little sense in expending resources collecting data to which the model is insensitive. This chapter evaluates the effects of soft inferences on solution times of the model, and at what values the soft inference weights begin to affect the solutions found. Also evaluated is a strategy whereby soft inferences are applied to the top_i variables, in addition to the bcl_{ci} variables.

For this series of tests, the soft inference rules described in Chapter III are implemented, and the models solved at various values of soft parameter weights. Table 3 identifies the various values of the parameter weights used in this testing phase.

	Scale								
Rule	1	1.5	2	2.5	3	5	7.5	12	20
CLLI	0.750	1.125	1.500	1.875	2.250	3.750	5.625	9.000	15.000
NPACOC	0.600	0.900	1.200	1.500	1.800	3.000	4.500	7.200	12.000
OCN	0.400	0.600	0.800	1.000	1.200	2.000	3.000	4.800	8.000
Equip "ESS"	0.580	0.870	1.160	1.450	1.740	2.900	4.350	6.960	11.600
Equip "DCO"	0.025	0.038	0.050	0.063	0.075	0.125	0.188	0.300	0.500
Equip "DMS250"	0.360	0.540	0.720	0.900	1.080	1.800	2.700	4.320	7.200
Equip "DMS100"	0.020	0.030	0.040	0.050	0.060	0.100	0.150	0.240	0.400
Equip "DMS10"	0.400	0.600	0.800	1.000	1.200	2.000	3.000	4.800	8.000

Table 3. Soft Parameter Weights Used in Testing

This table identifies the parameter values used in evaluating the effect of introducing soft inferences into the model. Values at a scaling of 1 are inherited from the CLIPS AI. The weights are scaled in order to maintain their relative proportions. Equipment rules "ESS," "DCO," and "DMS250" apply weights to the transit classes (3 and 4) of the node. "DMS100" and "DMS10" apply weights to the local exchange classes. Remaining rules are as described in Chapter III. When scaled by a factor of 20, most of the rules are sufficiently weighted to overrule any model assumption except *Top_Mesh* and *Min_Level*.

A. IMPACT OF SOFT INFERENCES ON SOLUTION TIME

Because soft inferences are heuristics, these rules may provide incorrect and contradictory indications of the actual class for some nodes. The preliminary testing of the baseline formulation shows that the IP's solution times are sensitive to the values of objective function parameter weights (see Figures 7 and 8). This section assesses the impact on solution times of inputting soft inference weights. How well each rule predicts the ground truth class of a node is not germane to this thesis—the rules will probably change as GCAT is implemented. Rather, the behavior of the model when contradictory and possibly infeasible inferences are applied is of interest.

Table 4 presents the correctness of an implementation of the soft inference rules described in Chapter III. A soft inference data point is "correct" if the associated rule applies a weight to the ground truth class of the node. Notice from Table 4 that many of the inferences generated by the soft rules incorrectly infer the actual class of the switching stations. While not obvious from Table 4, in many cases the various rules are contradictory, indicating a node is both a transit, and a local exchange. "Incorrect" soft inferences may or may not cause classification errors—the next section describes the effect of soft inferences on the solution.

The introduction of soft inference weights increases the IP's solution times. In the baseline model, the effect is considerable, in one case pushing solution time past the 600 second cut-off. In most cases, the worst effects are at high scaling values. Preprocessing routines greatly reduce this negative effect (see Table 5).

Network	CLLI		NPACOC		OCN		Equip	
	Correct	Incorrect	Correct	Incorrect	Correct	Incorrect	Correct	Incorrect
Net-0	1	2	0	1	9	2	4	6
Net-1	1	0	-	-	2	0	14	1
Net-2	1	2	-	-	5	0	17	3
Net-3	2	0	-	-	22	1	27	7
Net-4	2	0	-	-	22	1	28	6
Net-5	1	2	-	-	16	2	1	3
Net-6	3	0	-	-	16	2	11	7
Tracy	2	2	1	11	26	1	-	-
Balt	3	3	2	9	9	0	-	-

Table 4. Accuracy of an Implementation of Soft Inference Rules.

This table identifies the number of times soft inference rules correctly and incorrectly influence the class of nodes for the test networks derived from U.S. regional PSTNs. A "-" indicates no data was available for a particular soft inference rule. This implementation of soft inferences introduces many "incorrect" inferences into the model.

	Solution time in seconds								
	Baseline			Model A			Model B		
Network	No Softs	Worst	StdDev	No Softs	Worst	StdDev	No Softs	Worst	StdDev
Net-0	2.64	3.3	0.46	0.66	0.77	0.07	3.29	1.15	0.19
Net-1	7.19	12.79	2.59	0.71	0.83	0.06	0.6	1.54	0.25
Net-2	18.13	29.11	4.14	0.71	0.99	0.06	0.65	0.98	0.07
Net-3	5.5	9.06	0.96	0.77	0.77	0.04	0.65	1.65	0.25
Net-4	11.48	13.3	1.37	1.48	1.48	0.13	1.15	1.21	0.13
Net-5	36.52	124.96	33.47	1.05	17.58	4.54	1.04	5.71	1.11
Net-6	16.15	402.88	120.05	1.49	2.31	0.15	2.36	2.58	0.26
Tracy	50.2	230.58	65.98	1.32	1.64	0.13	1.15	1.38	0.11
Balt	59.59	>600	165.53	1.27	1.7	0.11	1.26	2.09	0.17

Table 5. Model Behavior with the Introduction of Soft Inferences

This table shows the variations in solution times as soft inference are introduced into three IP model variants. Each model is solved twice at each scaling of the soft inference weights identified in Table 3. "Model A" employs the *Leaf_Pluck*, *Min_Hop*(1.0), *Z_Loop* and *No_Eqn* preprocessing routines. "Model B" uses *Min_Hop*(0.1), *Leaf_Pluck*, and *Z_Loop*. The "No Softs" column displays the worst of the two times obtained with no soft inferences introduced in the model. The "Worst" and "StdDev" columns display the worst solution time and standard deviation of the solution times obtained with the introduction of soft inference weights, for both trials and across all nine scalings of the soft parameters (n=18). Notice the solution times of the baseline model are worsened considerably by the introduction of soft parameters. The application of preprocessing routines greatly reduces the worst observed solution times and moderates the variance.

B. ABILITY OF SOFT INFERENCES TO INFLUENCE THE SOLUTION

Soft inferences influence the solutions at relatively low weights. Table 6 indicates which network solutions are influenced by the introduction of soft inferences, and at what scaling value. Considering the relatively large parameter weights injected into the objective function, most network solutions are very resistant to change—an indication that the hierarchical structure of PSTNs is largely dictated by the inviolate *Top_Mesh* and *Min_Levels* considerations. Several insights about the behavior of the IP are gained:

	Scaling at which solution changes		
Network	Baseline	Model A	Model B
Net-0	3	3	3
Net-1	2		
Net-2	2	2.5	
Net-3			
Net-4			
Net-5	2.5	2.5	7.5
Net-6			
Tracy	1	1	1
Balt	1	1	1

Table 6. Scaling of Soft Inference Weights Yielding Alternate Solutions

Shown is the minimum scaling for soft parameters at which the solution found by the IP is modified. The values of the weights can be seen in Table 3 by referring to the appropriate "Scale" column. Notice the relatively low magnitudes at which soft inferences can influence the solution. The type of preprocessing affects how soft inferences influence the solutions found. The interaction of soft inferences and preprocessing is discussed in detail in the text.

- Soft inferences and the preprocessing routines interact. Net-1, with no preprocessing applied, is influenced by soft weights (equipment rule DMS250, specifically) to solve with node 5 at top-level status (see Figure 15 in Appendix A). Yet node 5 is a leaf node, and with the *Leaf_Pluck* preprocessing applied, this solution is prevented in models A and B. The situation with Net-2 is similar—leaf node 34 is influenced by an equipment rule to become a top-level node in the baseline model, a situation prevented

in models A and B. However, at higher scaling, soft weights influence node 31 to become a top in model A. Since $centrality_{31}$ = .32, the *Min_Hop*(0.1) preprocessing option of model B prohibits this solution. This illustrates that the solution speeds attained by the preprocessing routines are paid for with a loss of model generality. Restricting the model from finding solutions with top-level leaf nodes is probably a small price to pay for the speed advantage gained. However, when soft inferences seem to suggest an improbable hierarchy, it may be a clue that the logical topology, inferred from a physical network prior to invoking the IP, is incorrect. Over-restricting the model via preprocessing routines will obscure evidence of this nature.

- The soft inferences perform as envisioned. When contradictory inferences exist, stronger rules or combinations of rules overwhelm weaker ones, and soft inferences may over-rule model assumptions. At a weighting scale of 1, solutions for networks Balt and Tracy change. Tracy and Balt are both essentially trees, each with only one trio of nodes connected in a ring (see figures 20 and 21 in Appendix A). For Tracy, node 58 (the sole member of the triplet ring that is not a top) fires the CLLI, NPACOC, and OCN rules. The cumulative effect of this weighting is to force node 58 into top-level status (introducing a classification error, in this case). An analogous situation occurs in Balt—node 87, the sole non-top member of the triplet ring, is pushed to top-level status by the CLLI rule (correcting a misclassification of the baseline model). Notice that for Tracy's node 58, the soft inferences are contradictory. The rules suggesting this node is a transit exchange (CLLI and NPACOC) successfully overwhelm the OCN rule suggesting node 58 is a local exchange. The "ground-truth" configuration of Balt's node 87 violates the models' *Min_Tops* assumption.

The only way a correct classification of this node can occur is through the intervention of soft inferences. The soft inferences perform exactly as hoped in these networks.

- Soft inferences are able to effect considerable change in the IP's solution. The changes to the Net-5 solutions as a result of soft inferences are quite dramatic. At relatively low scale factor in the baseline and A models, soft inferences influence node 25 to attain top-level status (see Figure 18 in Appendix A). Because of the *Top_Mesh* requirement, for this to occur an existing top (node 21) must diminish to class 5 status, and fourteen descendents further diminish to class 6. This solution is prevented in model B, again because of node 25's high *centrality*. However, at sufficiently high scaling of the soft parameters in model B, node 21 is demoted anyway, primarily a result of the OCN rule. The structures resulting from soft inferences are quite unlikely, with fourteen added class 6 nodes, most in mesh configurations.

C. SOFT WEIGHTS FOR TOP-LEVEL NODES

The soft rules described previously operate on the surmised class(es) of a node. This section briefly evaluates an alternate method of implementing soft inferences by weighting the top_i variable for nodes i deemed to be at the top-level of the network. The implementation is simple, using just one additional soft data parameter in the objective function.

- $CTOP_i$ – a soft inference weight applied to influence the top-level status of node i in the IP's solution. Adding this weight modifies the objective function as shown:

Maximize

$$ZWT \cdot zclass - TWT \cdot \sum_i top_i + PWT \cdot \sum_i bcl_{5i} +$$

$$\sum_i \sum_c SOFT_{ci} \cdot (bcl_{ci}) + CTOP_i \cdot top_i \qquad (12)$$

Using model A previously described, Net-6 (previously insensitive to soft inferences) is successfully induced into an alternate structure by introducing $CTOP_{18} = 3$ and $CTOP_{19} = 3$ into the model (all other soft inferences are also introduced, at a scale factor of 1). As a more compelling example case, inspection of Net-5's topology shows it would not be unreasonable to expect node 24 to be at the top level of the network. Adding $CTOP_{24} = 1.1$ (less, with a conflicting OCN rule weight removed) is sufficient to effect this change.

The simple $CTOP_i$ tactic adds a potentially useful tool to soft inferences. In fact, the soft rules implemented for this testing do not provide an inference about the actual class of a node, but rather suggest whether a node is a transit or local exchange. These rules must therefore apply weights to multiple classes for each node, since a transit node could be either class 3 or class 4. Weighting the top_i variable for such nodes could simplify or augment soft rules addressing the surmised class of a node.

VIII. CONCLUSIONS AND RECOMMENDATIONS

The node classifier IP is suitable for service within GCAT. The IP produces acceptably accurate classifications for the U.S. regional PSTNs, and with the application of preprocessing, also returns timely solutions. The formulation is flexible enough to be tuned to seek a variety of potential hierarchical PSTN variants. This chapter describes the model recommended for implementation in GCAT. We close with a discussion of work still needed and a comparison with the alternate node-classification algorithm.

A. OPTIMAL FORMULATION

The node-classifier formulation recommended for implementation in GCAT is the IP incorporating the *Leaf_Pluck* and *Min_Hop*(1.0) preprocessing routines. We also recommend retaining an ability to implement the *Z_Loop* strategy. *Leaf_Pluck* and *Min_Hop*(1.0) are powerful routines, the primary agents responsible for reducing solution times to acceptable levels. The *Min_Hop* variant selected does not incorporate variable-fixing for non-top-level nodes because addition of this feature does not discernibly improve solution times. This feature does, however, add assumptions to the model that may restrict soft inferences from influencing the solution. The recommended model achieves swift solution speeds with minimum loss of model generality. Employing these preprocessing routines requires accepting only that a node with *centrality* = 0 is at the top network level, and that leaf nodes cannot be tops.

We recommend retaining the *Z_Loop* strategy, perhaps via a user-selectable switch, for several reasons. In cases where the fewest levels is overwhelmingly preferred, and the *Min_Hop*(1.0) and *Leaf_Pluck* routines are acceptable, the looping strategy is not necessary. However, the *Min_Hop*(1.0) or *Leaf_Pluck* routines may be

inappropriate, either in general for certain PSTN families, or because of their interaction with soft inferences. It also may be desirable to investigate alternate network structures with the *Min_Levels* assumption less strongly enforced. For these occasions, *Z_Loop* should be in place to speed solution times in the absence of the other routines, or to present alternative solutions for consideration by the analyst. In these cases, allowable values of *zclass* may be limited, if desired. In the absence of other preprocessing, *Z_Loop* significantly reduces solution time when seeking the solution using the minimum possible levels (see Table 15 in Appendix D). In concert with the other recommended preprocessing routines, *Z_Loop* can quickly present solutions over a range of *zclass* values (see Figure 14). The *Z_Loop* strategy adds considerable flexibility to the formulation.

Table 7 compares solution times of the recommended model (including *Z_Loop*) with those of the baseline formulation. Accepting the few additional assumptions of the preprocessing routines seems reasonable when weighed against the considerable improvement in solution times. These solution times, under three seconds for most networks, are clearly acceptable for a GCAT method.

B. CONTRAST WITH THE INTELLIGENT ENUMERATION ALGORITHM

The "Intelligent Enumeration" (IE) algorithm of J. Brandeau (1998), introduced in Chapter I, is a competitive solution technique to the node classification problem. In summary, this algorithm employs an all-pairs shortest path algorithm to identify all possible combinations of top-level nodes that could be present in a network formed with the fewest possible levels. It then uses a top-down classification scheme to assign node classes based on a node's minimum distance from the nearest top-level node. The optimal solution is determined by evaluating each classification scheme against an objective function similar to the IP's.

Network	Solution Time (seconds)	
	Recommended Formulation	Baseline Formulation
Net-0	0.73	3.54
Net-1	0.61	9.06
Net-2	0.82	19.12
Net-3	0.77	10.11
Net-4	1.19	11.48
Net-5	1.17	41.83
Net-6	1.38	19.06
Tracy	2.01	50.75
Balt	2.01	59.59
Lop4a	1.40	45.79
Lop4b	1.45	8.18
Lop5a	1.43	18.73

Network	Solution Time (seconds)	
	Recommended Formulation	Baseline Formulation
Lop5b	0.97	-
Lop-6	1.79	17.96
Lop6a	2.07	-
Lop6b	2.39	-
Net-5_6	3.57	107.43
Net-4_6	8.43	432.76
Net-3_6	11.87	511.30
HugeC	5.68	-
HugeB	12.47	-
HugeA	21.85	-
Huge	63.40	-

Table 7. Solution Times of the Recommended Model

Solution times for the formulation incorporating *Leaf_Pluck*, *Min_Hop*(1.0), and *Z_Loop* are in the column labeled "Recommended Formulation." The tabled values are the average value of three trials. For these trials, the *Z_Loop* routine exits upon obtaining the first solution, so times presented represent time to the first solution. For comparison, solution times of the baseline model are also presented. A "-" indicates no solution was obtained in 600 seconds. The test machine is a 166 MHz PC.

In comparing the mathematical programming and enumerative approaches, the clear speed advantage is to the IE algorithm. The speedy preprocessor code used in this thesis derives from the IE algorithm. Another advantage of the JAVA-language IE algorithm is that it does not require its users to own GAMS or the OSL solver. The IE algorithm can quickly present an analyst with many alternate solutions, rank-ordered using any conceivable fitness function. The IP can also present alternative solutions, but only at the expense of additional processing time. Lastly, the enumeration algorithm can implement non-linear soft inference functions, should any ever be devised.

However, the enumeration algorithm is very specifically coded to seek solutions of the fewest possible levels and with the fewest possible top-level nodes. In other words, it probably lacks flexibility in comparison with the IP. At this stage of GCAT's development, the node classification problem is not well defined, and the assumptions and requirements for the PSTN node classifier model are likely to evolve. A key concern about implementing the enumeration algorithm is that its workings are not as analytically accessible as the IP's, and a requirement to revise its implementation in the future may prove difficult or impossible.

C. SHORTCOMINGS AND SUGGESTIONS FOR FURTHER RESEARCH

The main shortcoming of this thesis is the small sample size—only nine test networks derived from real-world PSTNs. The IP is intended for use in analyzing networks not of U.S. origin, yet no logical topologies of overseas networks are included in the testing. Given the extensive experimentation done to optimize the IP's performance on this small PSTN sample population, it is quite possible the model is over-optimized. A clear requirement exists to re-validate the models' assumptions on the actual target population, non-U.S. PSTNs.

The modest number of test networks actually derived from real PSTNs also affects the quality of the soft inference testing. The test PSTNs solve more-or-less correctly without the application of soft inferences. Most errors present in the solutions of the baseline formulation are not addressed by any surmised soft rule. Consequently, the introduction of soft parameters into the model for these networks can only diminish the accuracy of the solutions found. Without a sizable sample of test networks that require soft inferences to solve "correctly," the conclusions about the performance of soft inferences are incomplete.

A further shortcoming of the IP in this application is that some form of preprocessing is required to speed solution times to acceptable levels. The added assumptions required for implementing the preprocessing routines of this thesis may be unsuitable in some applications. In particular, performance of the IP in the absence of the *Min_Hop*(1.0) would be marginally acceptable (see Figure 12). Finally, the selection of objective function weights and other model attributes are clearly tailored for classification of U.S. regional PSTNs. When the node classifier is applied to other families of PSTN, it is likely that these attributes will need to be revalidated. This implies a requirement for some baseline knowledge of the PSTNs being analyzed to establish appropriate penalty weights and develop or validate solution-accelerating enhancements. In other words, the IP is not "on-size-fits-all." It is unlikely this formulation can correctly classify foreign PSTNs without some adjustment of its parameters.

APPENDIX A. CHARACTERISTICS OF THE TEST NETWORKS

Contained within this appendix are descriptive characteristics of the test networks, and figures depicting the networks derived from U.S. region PSTNs.

Network	Location	# of Nodes	# of Arcs	# of Triplet Rings	# of Leaf Nodes	# Nodes in Triplet Rings	Max Node degree	Longest-Shortest Path	Lowest Class used
Networks derived from U.S. regional PSTNs									
Net-0	Notional	21	21	1	9	3	3	6	3
Net-1	Baltimore area	27	32	6	15	12	21	4	4
Net-2	Baltimore area	38	47	10	20	18	30	4	4
Net-3	Georgia	34	38	4	20	7	10	5	4
Net-4	Georgia	34	46	16	17	13	12	5	4
Net-5	D.C and N. VA	34	79	101	8	26	17	4	4
Net-6	D.C and N. VA	42	96	125	13	27	16	5	4
Tracy	California	90	90	1	70	3	31	5	4
Balt	Baltimore area	103	103	1	87	3	66	5	4
Large Notional Networks									
5_6	Aggregation	76	183	242	22	53	21	5	4
4_6	Aggregation	110	247	321	39	66	24	5	4
3_6	Aggregation	144	320	472	56	79	27	5	4
HugeC	Aggregation	118	313	485	36	78	25	5	4
HugeB	Aggregation	152	402	664	44	106	27	5	4
HugeA	Aggregation	220	575	1290	78	132	33	5	4
Huge	Aggregation	304	948	2912	88	212	39	5	4
Networks with modified Longest-Shortest Paths									
Lop4a	Modified Net-4	35	47	16	17	13	12	6	3
Lop4b	Modified Net-4	36	48	16	17	13	12	7	3
Lop5a	Modified Net-5	35	81	101	8	26	17	5	4
Lop5b	Modified Net-5	36	82	101	8	26	17	6	3
Lop6	Modified Net-6	41	95	125	14	27	16	5	4
Lop6a	Modified Net-6	43	97	125	14	27	16	6	3
Lop6b	Modified Net-6	44	98	125	14	27	16	7	3

Table 8. Test Network Characteristics.

These summary statistics are intended to give a snapshot view of the key characteristics of the test networks. A "triplet ring" refers to a trio of nodes that are completely connected. The number of triplet rings and nodes involved in triplet rings are intended to provide an indication of the degree to which a network contains mesh topologies. Generally, solution times increase as the number of nodes, arcs, and number of nodes involved in mesh configurations, increases.

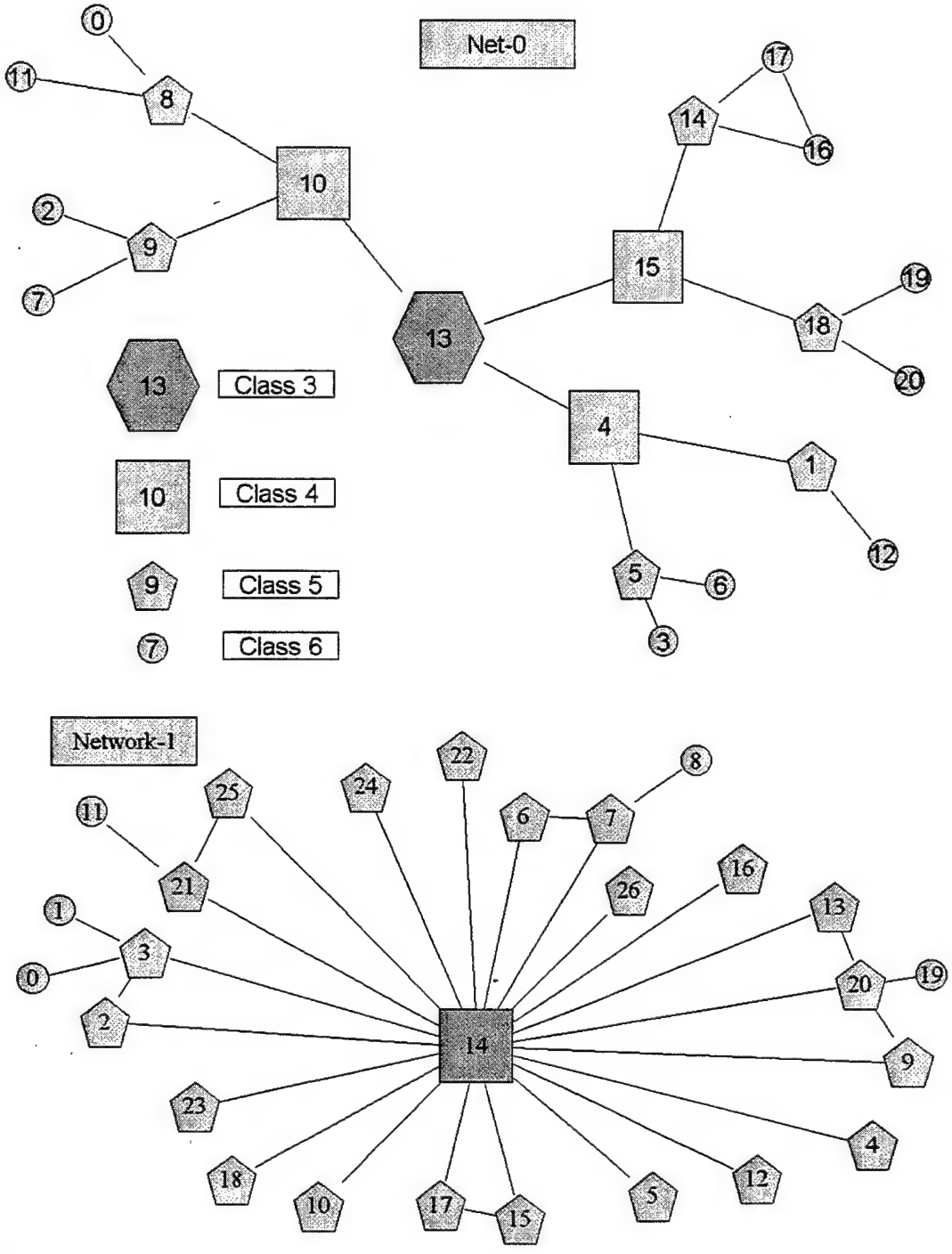

Figure 15. Logical Structures of Test Networks 0 and 1

Network-0's topology is notional, and the configuration of nodes 16 and 17 (interconnected class 6) is atypical. Network-1 is derived from a Bell Atlantic regional PSTN in Maryland.

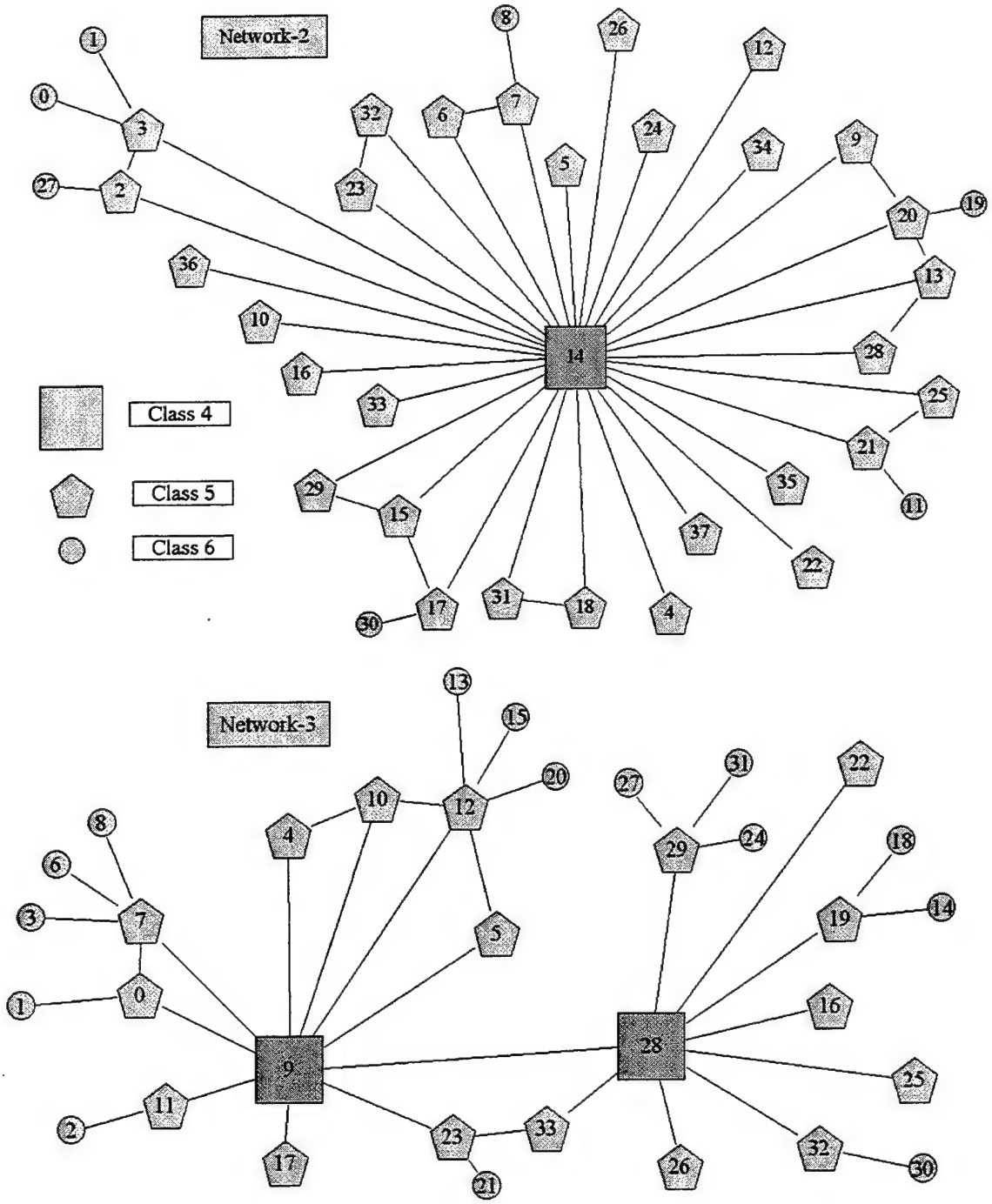

Figure 16. Logical Structures of Test Networks 2 and 3

Network 2 is derived from a regional PSTN serving the Baltimore, Maryland, region. Network 3 is located in rural Georgia.

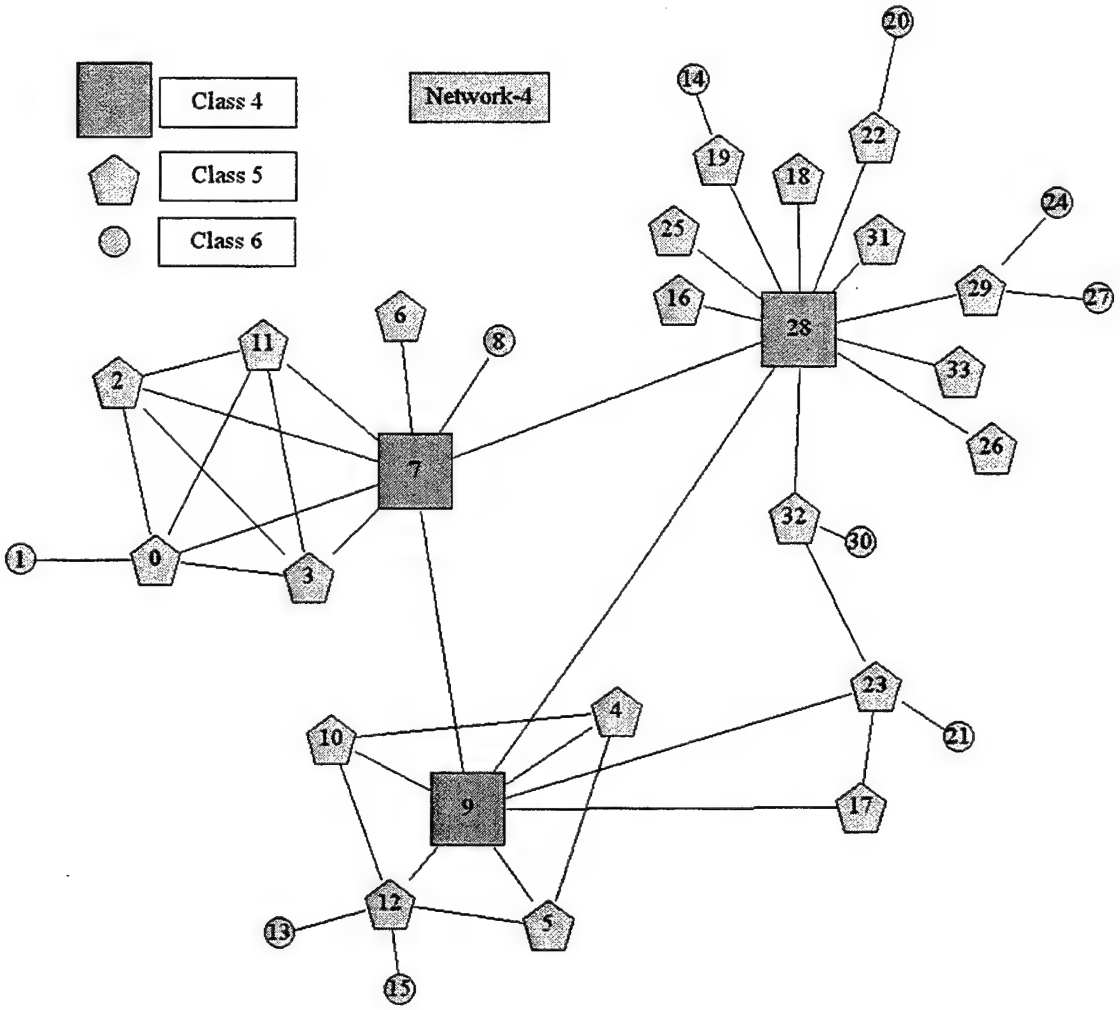

Figure 17. Logical Structure of Test Network 4

This network is derived from a portion of the Southern Bell regional PSTN located in rural, southeastern Georgia. It is derived from the same physical network as test network 3. Node 8 solves incorrectly as a class 5 because it violates the models' *More_5s* assumption.

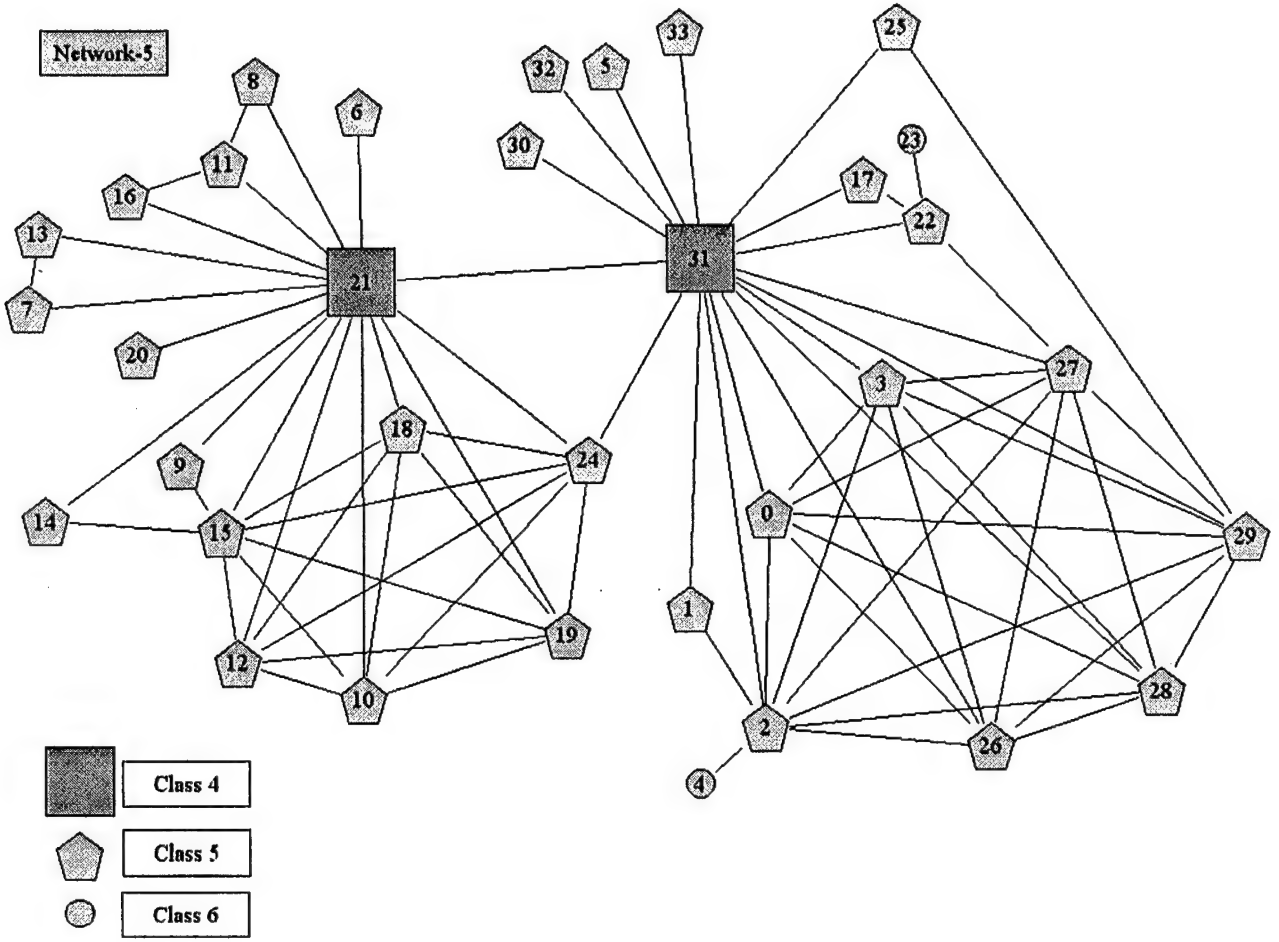

Figure 18. Logical Structure of Test Network 5

This network is derived from the PSTN serving Washington, D.C., northern Virginia, and portions of Maryland.

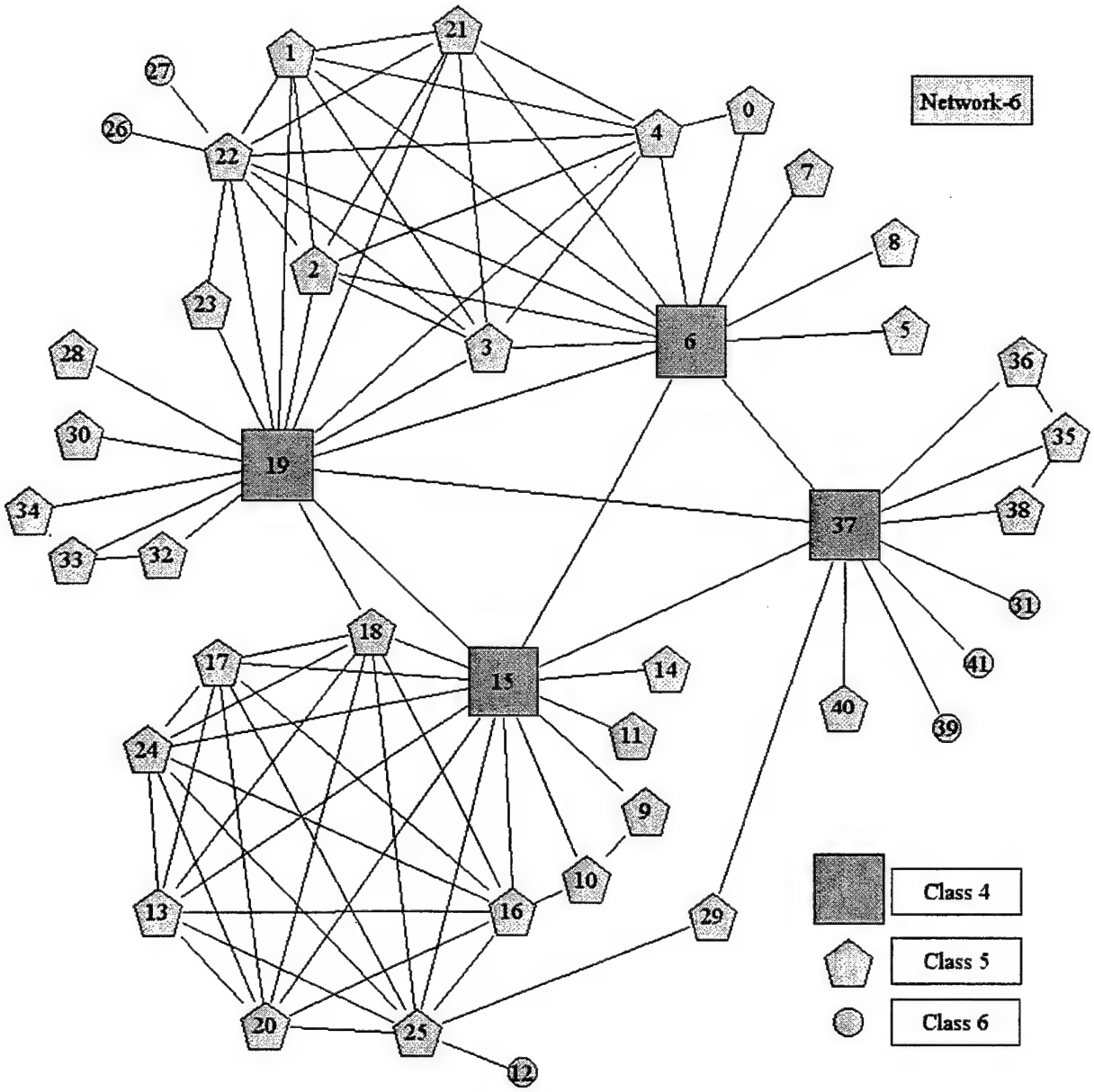

Figure 19. Logical Structure of Test Network 6

This network is an alternate derivation of the PSTN from which test network 5 was derived. Nodes 31, 39, and 41 solve incorrectly as class 5 (they violate the *More_5s* model assumption).

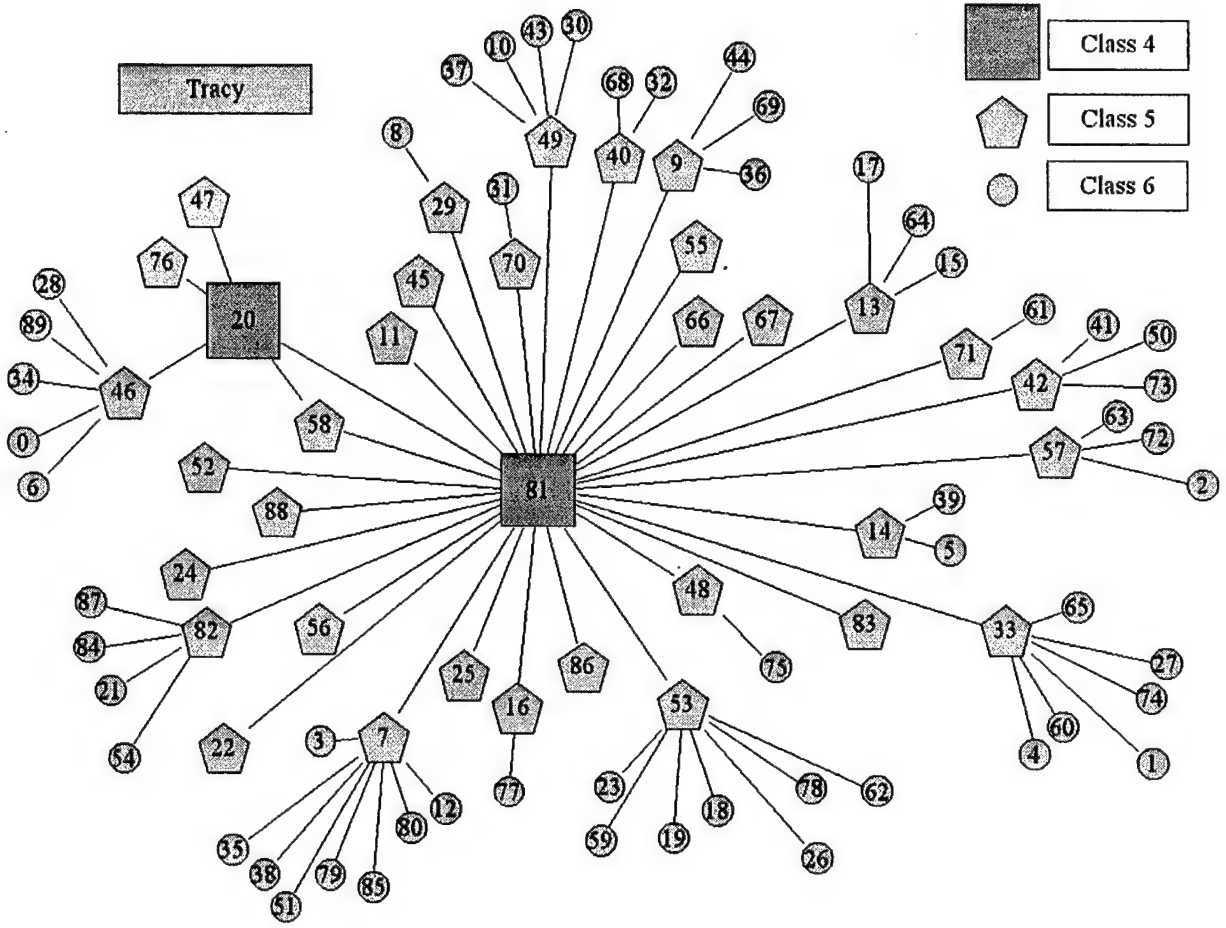

Figure 20. Logical Structure of Test Network Tracy

This network is derived from a regional PSTN in California.

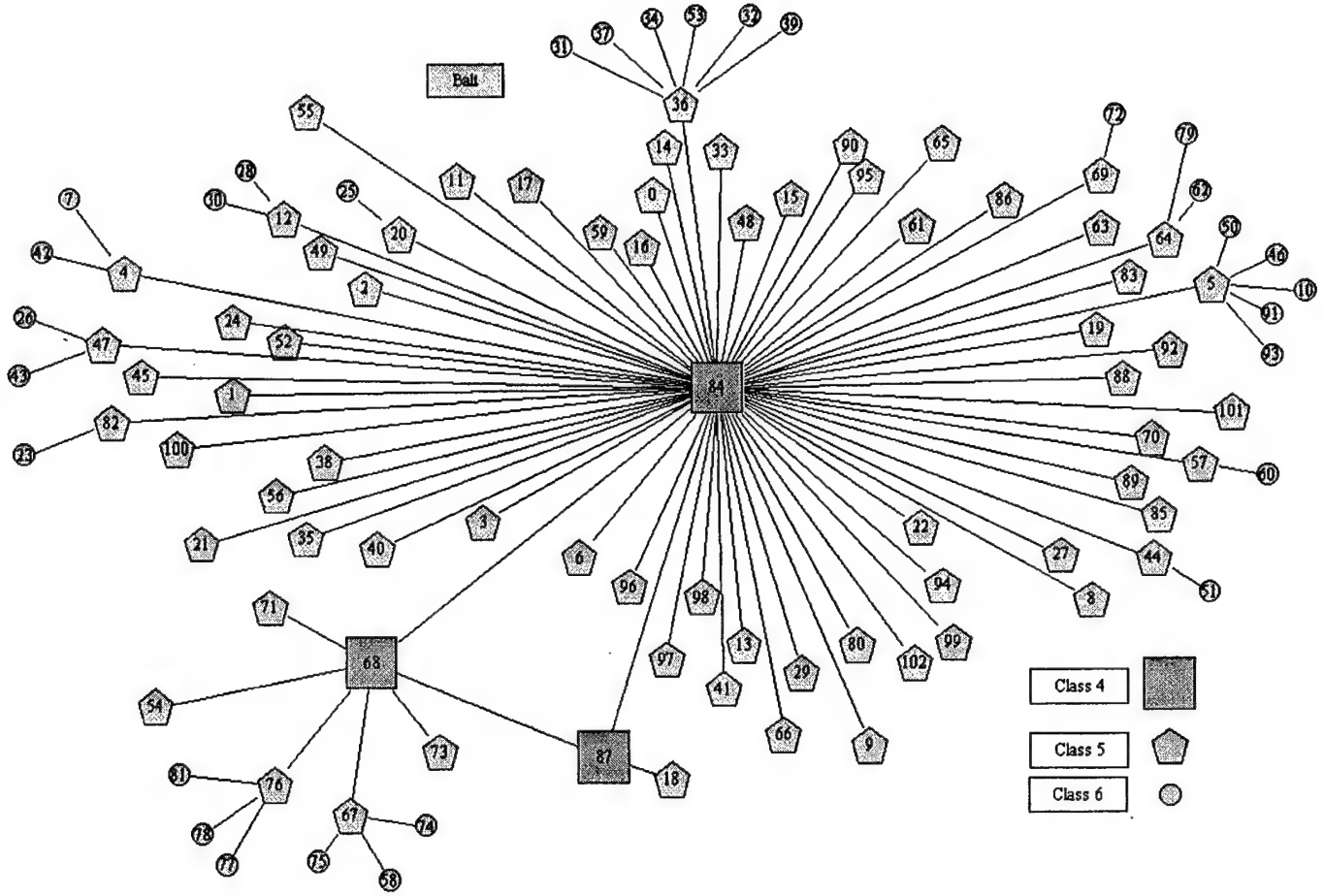

Figure 21. Logical Structure of Test Network Balt

This network is derived from a PSTN in Maryland. Nodes 87 and 18 are incorrectly classified by the IP as class 5 and 6, respectively.

APPENDIX B. PRELIMINARY TESTING OF THE BASELINE FORMULATION

This appendix contains the data obtained during preliminary testing of the IP.

	Value of ZWT												
Network	***100***	***90***	***80***	***70***	***60***	***50***	***40***	***30***	***20***	***10***	***7***	***4***	***1***
Net-0	0.88	0.71	0.99	0.66	0.71	0.66	0.66	0.66	0.66	0.635	0.66	0.61	0.55
Net-1	2.08	2.47	2.86	2.86	3.07	2.52	2.08	1.48	2.75	3.185	4.67	0.6	0.55
Net-2	4.62	4.55	4.4	4.67	4.56	4.39	4.45	4.56	4.39	3.05	1.15	0.99	0.99
Net-3	1.64	1.98	1.64	1.65	1.65	1.7	1.71	1.64	1.7	1.595	1.43	1.43	3.24
Net-4	3.3	3.46	3.19	3.19	3.14	3.19	3.19	3.13	3.24	3.19	3.02	2.41	5.71
Net-5	11.69	5.61	9.66	167.63	6.2	15	20.87	16.81	11.92	3.46	3.08	0.88	1.48
Net-6	4.61	4.51	4.84	7.25	4.06	4.56	4.5	4.17	4.56	5.05	5.11	3.4	105.07
Tracy	14.94	14.5	15.1	16.42	14.83	14.44	15.21	15.05	14.94	19.72	25.16	11.31	18.01
Balt	18.4	18.73	18.51	19.39	20.43	19.93	20.21	19.06	16.8	15.52	14.39	13.73	19.27
Lop4a	14.39	13.02	38.34	15.55	13.02	17.85	12.63	15.38	14.01	20.13	19.11	11.21	13.35
Lop5a	4.18	3.96	4.01	3.02	3.46	3.35	3.3	3.19	3.46	3.18	5.05	2.09	7.03
Lop5b	11.59	9.39	108.69	343.72	7.14	15.15	103.86	600	17.3	9.04	8.13	301.93	9.72
Lop-6	3.68	7.25	3.79	3.9	7.91	14.39	4.17	3.51	3.52	3.52	6.54	2.52	4.45
Lop6a	390.19	103.1	435.39	326.86	159.34	125.34	321.32	263.47	407.66	103.89	89.25	110.18	17.9
Lop6b	166.48	454.95	318.95	415.3	248.54	407.1	81.84	432.26	261.67	469.145	102	35.48	43.78
Net-5_6	122.65	51.47	33.83	52.57	27.02	45.7	30.21	86.23	261.28	10.38	10.76	10.22	47.46
Net-4_6	75.36	140.11	138.58	72.28	74.86	75.25	137.81	72.72	134.4	58.245	58.22	205.09	296.54
Net-3_6	165.88	249.04	152.25	203.72	153.24	170.43	150.83	148.68	149.78	223.32	-	-	-
HugeC	-	395.57	290.44	209.59	177.41	-	165.49	166.8	353.72	73.85	37.24	315.38	101.34
# values w/in 20% of best	2	1	2	2	5	3	4	3	3	5	8	14	5

Table is continued on the next page

	Value of TWT												
Network	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0.7**	**0.4**	**0.1**
Net-0	0.99	0.71	0.60	0.66	0.66	0.66	0.66	0.55	0.66	0.69	0.66	0.60	0.60
Net-1	3.79	2.42	1.92	4.94	1.98	2.08	4.62	3.57	3.79	2.50	1.70	1.54	1.59
Net-2	4.23	5.11	5.00	5.00	6.54	5.55	4.67	4.39	4.67	5.63	4.94	4.73	7.03
Net-3	1.48	1.70	1.54	1.48	1.81	1.70	1.65	1.65	1.76	2.09	1.75	1.76	2.30
Net-4	3.62	5.66	3.57	4.61	3.68	3.68	4.67	4.12	3.79	4.01	3.41	3.46	3.13
Net-5	19.01	22.30	16.81	19.38	22.02	23.84	16.42	14.39	26.75	7.72	9.23	2.70	31.08
Net-6	574.57	566.50	352.45	424.57	428.09	390.47	601.37	414.09	185.42	54.46	103.53	30.49	19.17
Tracy	5.47	6.43	12.11	10.38	12.19	14.20	13.43	21.07	16.15	14.50	19.88	17.47	28.21
Balt	11.29	7.67	7.42	8.57	9.50	22.11	16.40	22.47	23.79	20.35	21.07	22.25	22.79
Lop4a	15.60	9.83	11.37	9.83	12.47	15.71	16.32	12.75	12.91	16.48	15.71	19.00	11.31
Lop4b	1.38	2.30	1.70	2.31	2.25	1.15	1.32	1.32	1.43	1.81	1.20	1.26	1.15
Lop5a	18.95	10.33	16.37	59.48	20.16	11.20	368.11	79.97	10.88	5.00	2.80	3.51	8.51
Lop5b	13.68	43.88	46.37	11.26	14.39	79.53	22.49	63.55	14.67	20.59	12.50	43.58	9.36
Lop-6	573.43	250.90	139.95	120.81	134.33	97.93	89.59	264.31	239.40	12.01	10.17	3.82	6.68
Lop6a	543.81	526.16	567.19	469.20	567.44	571.17	337.43	572.02	419.23	442.47	171.42	470.86	137.26
Lop6b	587.45	584.41	514.60	488.12	-	483.87	529.21	-	530.12	549.62	240.44	187.90	492.50
Net-5_6	-	-	-	-	-	-	-	-	94.47	97.88	69.65	38.67	108.14
# values w/in 10% of best	3	2	3	2	0	1	0	2	0	0	3	7	7

Table 9. Solution Speeds as *ZWT* and *TWT* are Varied

Base parameter values are *ZWT*=100, *TWT*=.5 and *PWT*=.2. Unshaded cells are the times summed in the last row of each table. A "-" indicates no solution was obtained in 600 seconds. Tabled values are solution times in seconds on the 166 MHz PC.

Network	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
Net-0	14.17	14.78	2.20	1.65	2.74	1.93	2.53	2.74	2.20	2.69	2.59	2.47	3.24	2.52	3.03	3.30	2.25	2.37	2.97	1.93
Net-1	2.63	4.27	20.94	10.82	5.82	5.88	7.36	8.40	9.11	7.69	15.10	15.82	10.35	6.42	5.72	5.71	22.30	19.99	61.46	62.95
Net-2	10.93	7.96	58.00	27.24	15.98	15.60	14.55	16.31	21.91	16.53	46.36	44.32	32.74	17.20	16.81	16.48	56.41	52.78	34.16	32.91
Net-3	10.11	9.61	5.93	4.72	5.43	5.44	5.00	5.27	6.86	5.55	5.55	5.17	6.21	14.72	7.64	8.07	8.68	4.67	5.27	5.16
Net-4	12.25	200.53	7.64	6.92	10.82	10.71	10.11	10.82	13.51	11.15	11.20	10.60	13.71	15.93	15.93	15.27	16.75	6.97	11.04	10.99
Net-5	47.73	92.44	106.45	20.98	64.10	64.75	38.78	42.84	63.00	46.75	66.29	60.80	79.97	69.37	148.13	92.99	112.76	94.25	563.92	566.89
Net-6	254.91	234.38	6.53	5.71	3.90	4.11	3.79	4.07	8.90	4.17	4.34	3.96	8.51	6.80	6.32	5.32	6.16	5.72	4.01	4.01
Tracy	10.33	6.59	36.91	12.58	19.00	18.13	26.31	28.23	33.34	28.17	86.02	80.35	31.70	26.42	46.68	89.75	38.72	36.41	-	-
Balt	31.92	56.96	17.52	11.48	11.31	15.88	10.54	11.37	15.87	12.31	51.63	47.68	26.97	32.30	35.81	25.82	19.05	17.41	63.88	63.77
Lop4a	93.76	-	173.84	-	50.64	56.41	-	-	321.26	225.47	217.39	200.15	40.73	254.47	135.34	53.82	239.20	167.52	145.84	137.75
Lop4b	-	23.72	15.82	30.04	13.56	13.18	12.14	12.74	17.46	13.84	51.85	14.28	21.73	19.22	19.39	21.97	20.65	15.87	225.96	226.35
Lop5a	26.37	96.84	42.34	43.61	78.38	375.75	160.83	161.86	27.79	21.32	69.92	135.88	52.86	45.54	27.08	44.93	40.16	36.58	-	-
Lop5b	-	-	-	-	-	478.19	475.38	501.58	-	-	-	-	-	-	-	-	-	-	-	-
Lop-6	-	-	-	-	-	517.62	310.17	328.68	479.91	388.43	-	-	506.82	-	-	-	-	-	-	-
Lop6a	142.31	113.30	54.21	49.87	53.22	54.88	50.86	53.94	92.61	55.25	56.80	55.75	63.58	70.96	82.44	71.78	54.38	48.66	53.60	53.05
Lop6b	60.86	-	141.43	56.63	62.84	62.17	59.60	62.95	108.43	71.40	74.42	62.29	78.66	83.33	87.71	87.00	67.66	56.29	61.79	62.18
Net-5_6	188.50	-	-	266.56	87.77	87.16	83.10	86.29	115.61	88.81	474.89	84.09	376.43	91.84	93.70	92.66	-	528.10	-	-
Net-4_6	430.90	531.19	372.83	202.62	383.54	252.21	245.18	253.43	314.06	258.32	266.98	251.56	296.66	-	307.75	304.18	370.03	265.02	254.80	256.94
Net-3_6	-	-	-	-	-	515.37	520.20	523.55	-	-	-	523.33	-	-	-	-	-	-	-	-
HugeC	-	-	-	-	-	529.26	522.29	530.96	516.24	462.91	-	524.93	-	512.73	505.70	484.93	-	-	-	-
HugeB	-	440.67	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
HugeA	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
HugeNet	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
# low values*	3	3	3	8	8	11	12	12	1	11	3	9	1	2	3	2	2	4	5	6

Series	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
bbpreproc	0	0	1	2	2	2	2	2	2	2	2	2	3	3	3	3	4	4	4	4
strategy	1	1	1	1	2	2	8	8	8	8	16	32	1	2	8	32	1	1	4	32
presolve	-1	0	0	0	-1	0	0	1	2	3	3	0	0	3	1	1	0	0	0	0

Table 10. Effect of Solver Options on Solution Times

A "-" indicates no solution was obtained in 600 seconds of processing time. * This row provides a count of the solution times for the series that are within 30% of the minimum for all series on each network. These values are unshaded in the main table.

	Series								
Network	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**
Net-0	1.80	1.77	2.8	1.87	2.63	1.87	2.69	1.98	3.23
Net-1	8.79	8.85	9.89	9.12	13.51	7.2	11.04	11.32	9.18
Net-2	17.13	16.05	17.14	16.15	14.39	15.33	16.15	16.04	15.98
Net-3	5.38	5.6	5.44	5.66	4.89	5.27	5.61	5.6	5.23
Net-4	11.15	11.25	11.26	11.26	10.16	10.43	14.39	11.64	10.43
Net-5	43.17	69.1	78.1	133.57	-	27.3	103.48	46.57	46.46
Net-6	16.86	93.82	142.26	-	-	115.29	16.58	17.41	4.34
Tracy	52.84	72.39	91.73	51.9	51.08	53.17	52.01	54.87	62.39
Balt	62.78	64.97	100.13	63.11	59.82	61.29	62.18	63.33	14.72
Lop4a	50.26	50.2	49.37	52.01	41.69	40.21	94.47	51.19	-
Lop4b	4.61	4.66	4.78	4.73	3.68	3.96	7.63	4.67	64.98
Lop5a	15.54	15.87	18.02	15.71	10.65	11.04	28.5	15.49	87.55
Lop5b	42.62	-	-	-	50.53	-	67.88	42.4	469.39
Lop-6	12.86	70.48	85.02	-	-	12.97	26.48	13.35	140.73
Lop6a	-	-	-	-	-	505.7	602	-	50.15
Lop6b	582.38	161.26	192.46	-	-	516.52	-	-	60.74
Net-5_6	416.17	-	303.02	-	-	85.63	-	-	-
Net-4_6	267.33	263.38	384.14	265.24	257.66	262.54	264.08	268.47	-
Net-3_6	528.38	541.18	-	585.06	528.11	513.11	549.31	-	-
HugeC	-	-	-	-	-	529.92	-	-	-
# of values within 10% of best attained	7	3	0	3	9	15	3	4	6

Series	Priority
A	GAMS default (none)
B	sumtops, top
C	sumtops, top, p
D	p, top, sumtops, zclass
E	p, top, bcl_3, bcl_4, bcl_5
F	zclass
G	sumtops
H	p
I	zclass, sumtops, top

Table 11. Effect of Branching Strategy on Solution Times.

Unshaded values in the table indicate the solution time is within 10% of the best attained by any strategy on that network. A "-" indicates an optimal solution was not found in 600 seconds. In the legend, read the "Priority" column from left to right, e.g., for series C, *sumtops* was assigned the highest branching priority, followed by top_i, etc.

APPENDIX C. PARAMETER VALUES OF TOP LEVEL CANDIDATE NODES

Network	*Top-level Candidates*	*Degree Ranking*	*TotalHop Ranking*	*Centrality*	*MaxHops*	*In a Center Position on a L-S Path ?*
Net 0	13	Highest	Minimum	0.000	3	Yes
Net 1	14	Highest	Minimum	0.000	2	Yes
Net 2	14	Highest	Minimum	0.000	2	Yes
Net 3	9	Highest	Minimum	0.000	3	Yes
	28	2nd Highest	2nd Minimum	0.029	3	Yes
Net 4	7	2nd Highest	3rd Minimum	0.059	3	Yes
	9	2nd Highest	2nd Minimum	0.029	3	Yes
	28	Highest	Minimum	0.000	3	Yes
	32 *	10th Highest	3rd Minimum	0.088	4	Yes
Net 5	21	2nd Highest	2nd Minimum	0.029	3	No
	31	Highest	Minimum	0.000	2	Yes
	27 *	4th Highest	4th Minimum	0.088	3	Yes
Net 6	6	3rd Highest	2nd Minimum	0.024	3	Yes
	15	2nd Highest	2nd Minimum	0.024	3	Yes
	19	Highest	Minimum	0.000	3	Yes
	37	4th Highest	4th Minimum	0.071	3	No
	18 *	6th Highest	5th Minimum	0.095	3	Yes
Tracy	20	4th Highest	2nd Minimum	0.011	3	Yes
	81	Highest	Minimum	0.000	3	Yes
	58 *	14th Highest	5th Minimum	0.056	3	No
Balt	68	2nd Highest	2nd Minimum	0.010	3	Yes
	84	Highest	Minimum	0.000	3	Yes
	87	7th Highest	3rd Minimum	0.019	3	No

Table 12. Parameter Values of Top-level Candidate Nodes

This table presents the parameter values of nodes that are top-level candidates in the test networks derived from actual PSTNs. Nodes not in reality at the top-level are asterisked. A node enters the set of candidates if it is of highest degree in the network, has minimum *totalhops*, is on a position in a L-S Path which would indicate a top if the path were symmetric, or is one of the set of minimum *maxhop* nodes.

APPENDIX D. RESULTS OF PREPROCESSING TESTS

This appendix tables the data resulting from tests of preprocessing routines designed to accelerate solution times.

		Min_Hop(α)											
Network	Baseline	1.0	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	Leaf_Pluck	Class_6
Net-0	116.60	29.55	29.55	29.55	29.55	29.55	29.55	29.55	29.55	29.55	0.25	89.75	75.76
Net-1	53.15	27.40	27.40	27.40	27.40	27.40	27.40	27.40	27.40	27.40	0.00	27.40	26.35
Net-2	49.85	26.20	26.20	26.20	26.20	26.20	26.20	26.20	26.20	26.20	0.00	26.20	24.55
Net-3	52.95	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	12.10
Net4	52.90	0.53	0.53	0.53	0.53	0.53	0.53	0.53	0.53	0.53	0.53	13.75	14.91
Net-5	52.60	27.05	27.05	27.05	27.05	27.05	27.05	27.05	27.05	0.00	0.00	27.05	71.02
Net-6	51.40	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	26.90	71.18
Tracy	41.05	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	17.14
Balt	31.55	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	22.30
Lop4a	112.05	0.57	0.57	0.57	0.57	0.57	0.57	0.57	0.57	0.57	0.57	70.10	72.14
Lop4b	111.90	0.57	0.57	0.57	0.57	0.57	0.57	0.57	0.57	0.57	0.57	59.05	12.51
Lop5a	52.95	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	27.60	9.50
Lop5b	111.80	29.25	29.25	29.25	29.25	29.25	29.25	29.25	29.25	0.00	0.00	86.70	68.17
Lop-6	51.55	26.90	26.90	26.90	26.90	26.90	26.90	26.90	26.90	26.90	0.08	27.05	68.53
Lop6a	110.05	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	85.80	9.05
Lop6b	109.90	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	76.53	1.52
Net-5_6	44.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	23.90	4.40
Net-4_6	36.90	0.53	0.53	0.53	0.53	0.53	0.53	0.53	0.53	0.53	0.53	10.85	1.12
Net-3_6	29.80	1.07	1.07	1.07	1.07	1.07	1.07	1.07	1.07	1.07	1.07	1.07	0.00
HugeC	35.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	20.80	0.00
HugeB	28.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	17.80	0.00
HugeA	18.88	1.07	1.07	1.07	1.07	1.07	1.07	1.07	1.07	1.07	1.07	1.07	0.00
Huge	0.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	0.00	0.00	0.00

Table 13. Gap Between the Relaxed and Optimal Objective Function Values

The Baseline series has no preprocessing. The other routines are as described in the chapter on preprocessing.

Network	Baseline	Min_Hop(α) 1.0	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	Leaf_Pluck	Class_6
Net-0	3.54	1.93	1.98	1.10	1.10	2.09	3.90	1.15	1.21	1.09	1.10	1.95	1.60
Net-1	9.06	2.37	2.75	2.58	2.63	2.47	4.07	2.97	1.76	1.81	1.10	1.84	7.58
Net-2	19.12	5.60	5.71	5.82	5.99	5.28	6.70	2.81	2.86	2.59	1.42	3.08	8.95
Net-3	10.11	1.10	1.59	1.59	1.54	1.43	1.65	1.81	1.81	2.41	2.08	1.24	5.55
Net4	11.48	2.38	2.63	2.36	2.81	2.58	3.30	2.80	2.80	2.69	2.75	3.02	5.38
Net-5	41.83	4.05	4.39	4.61	4.61	4.45	5.82	5.11	4.78	3.57	2.47	4.10	21.75
Net-6	19.06	6.64	4.01	4.23	4.12	3.85	5.27	3.62	5.99	5.44	4.01	18.59	42.62
Tracy	50.75	4.18	4.34	4.73	5.50	5.22	6.70	5.54	6.38	5.93	4.06	3.52	44.87
Balt	59.59	5.87	6.09	6.70	9.17	10.06	11.81	10.33	10.49	10.65	5.22	4.94	56.79
Lop4a	45.79	1.70	2.31	2.25	2.26	1.98	4.40	2.09	2.04	2.25	2.26	11.35	23.51
Lop4b	8.18	1.81	2.09	2.14	2.25	2.36	2.64	2.58	2.36	2.30	1.97	2.66	4.11
Lop5a	18.73	2.31	2.58	3.13	3.02	2.91	3.24	2.53	2.53	2.64	2.20	7.75	5.44
Lop5b	-	2.69	3.07	3.90	3.90	3.24	3.73	4.07	4.40	2.36	2.36	137.04	16.04
Lop-6	17.96	8.01	6.32	8.95	5.55	7.91	9.00	7.03	7.08	7.14	3.96	6.84	53.82
Lop6a	-	4.05	4.45	4.45	4.17	4.28	4.45	4.40	5.55	4.39	4.07	171.73	50.37
Lop6b	-	3.74	4.45	4.62	4.50	4.29	4.78	4.06	3.84	3.95	5.00	-	70.31
Net-5_6	107.43	9.95	8.29	16.87	10.16	10.11	12.74	10.10	9.62	11.92	10.00	90.47	306.76
Net-4_6	432.76	16.15	17.63	18.68	18.29	18.01	25.16	20.81	20.43	18.08	15.38	48.20	117.88
Net-3_6	511.30	24.00	29.99	30.64	36.80	30.27	39.16	31.26	30.21	61.79	29.77	182.08	83.32
HugeC	-	20.05	43.89	27.63	22.19	25.10	27.07	26.26	22.08	23.13	20.98	92.58	43.72
HugeB	-	26.75	36.75	37.46	35.05	36.96	55.64	56.85	35.76	30.98	33.06	199.03	97.22
HugeA	-	65.53	70.47	79.81	149.45	75.19	151.82	79.31	86.78	108.53	88.54	544.53	186.69
Huge	-	140.01	NA	NA	NA	NA	NA	NA	NA	NA	140.34	129.68	584.51

Table 14. Solution Times for Various Preprocessing Routines

The Baseline series has no preprocessing. The other routines are as described in the chapter on preprocessing. "NA" values were not collected because of equipment failure. Cells containing a "-" indicate no optimal solution was returned within the 600-second time limit.

Network	zclass 4	3	2	1	Baseline (No Z_Loop)
Net-0	NA	1.43	5.98	13.07	3.54
Net-1	1.43	9.28	62.45	420.62	9.06
Net-2	2.64	16.76	552.45	-	19.115
Net-3	3.24	13.46	27.62	267.38	10.11
Net-4	3.89	54.43	149.78	-	11.48
Net-5	2.75	65.31	-	-	41.825
Net-6	7.03	26.15	-	-	19.055
Tracy	46.31	41.91	106.83	113.03	50.75
Balt	13.07	51.13	-	223.22	59.59
Lop4a	NA	39.6	-	-	45.785
Lop4b	NA	7.47	227.89	-	8.18
Lop5a	4.12	34.55	342.35	-	18.73
Lop5b	NA	34.71	172.97	-	-
Lop-6	6.38	9.33	309.78	-	17.96
Lop6a	NA	-	-	-	-
Lop6b	NA	317.25	-	-	-
Net-5_6	22.08	44.05	-	-	107.43
Net-4_6	72.77	-	-	-	432.76
Net-3_6	160.99	-	-	-	511.295
HugeC	160.99	-	-	-	-
HugeB	137.75	-	-	-	-
HugeA	-	-	-	-	-
Huge	-	-	-	-	-

Table 15. Solution Times for *Z_Loop* Strategy in the Baseline Formulation

This table depicts the solution times (in seconds) attained for each value of *zclass* between class 4 and class 1 using *Z_Loop* with no additional preprocessing. "NA" indicates that the value of *zclass* is infeasible for the network. A "-" indicates no optimal solution was attained in 600 seconds of processing. For the naïve implementation, the solution times appear to increase exponentially as *zclass* is fixed farther from its maximum feasible value. The time to the first solution is considerably improved over the naïve formulation without implementing the *Z_Loop* routine.

	CPU seconds					
Network	zclass=4	zclass=3	zclass=2	zclass=1	zclass=0	No Z_Loop
Net-0	NA	2.03	1.04	0.99	1.05	1.38
Net-1	0.88	0.82	0.83	0.82	0.76	1.37
Net-2	2.09	1.05	1.76	1.1	2.85	1.59
Net-3	2.09	1.04	1.1	1.04	1.1	1.48
Net-4	1.7	2.25	1.27	3.19	1.26	2.36
Net-5	2.25	1.98	1.7	2.25	1.87	2.47
Net-6	3.46	2.42	2.04	1.92	1.81	4.34
Tracy	2.74	2.47	1.98	2.53	1.87	2.02
Balt	3.13	3.02	3.24	3.23	2.53	2.31
Lop4a	NA	1.81	3.07	3.84	3.74	2.02
Lop4b	NA	1.87	3.57	3.4	3.85	2.04
Lop5a	1.92	1.7	1.7	1.76	1.59	2.04
Lop5b	NA	1.86	1.81	1.86	1.65	2.25
Lop-6	3.35	2.19	2.14	1.82	2.15	3.73
Lop6a	NA	3.46	1.97	2.31	4.34	3.57
Lop6b	NA	3.74	2.14	2.31	8.84	3.84
Net-5_6	8.68	16.87	9.95	4.61	5.16	6.49
Net-4_6	13.13	23.12	21.15	9.88	29.11	10.33
Net-3_6	21.53	49.88	34.54	35.59	43.61	16.53
HugeC	21.31	51.9	9.78	32.73	38.77	13.57
HugeB	29.05	43.23	19.28	16.2	14.5	40.59
HugeA	60.75	87.44	77.39	91.51	104.02	44.22
Huge	139.4	254.63	188.5	65.19	84.86	102.72

Table 16. Solution Times for the *Z_Loop* Strategy with Additional Preprocessing

This table shows the solution times obtained by *Min_Hop*(0.1), *Leaf_Pluck*, and *No_Eqn* preprocessing, in concert with *Z_Loop*. "NA" indicates the value of *zclass* is infeasible for the network.

LIST OF REFERENCES

Ahuja, R., Magnanti, T., and Orlin, J., *Network Flows--Theory, Algorithms, and Applications*, Prentice Hall, Inc., 1993.

Ash, G., *Dynamic Routing in Telecommunications Networks*, McGraw Hill, 1998.

Brandeau, J., *A Fast Algorithm for Classification of PSTN Switching Stations*, Draft M.S. Thesis in Operations Research, Naval Postgraduate School, Monterey, CA 93940, September 1998.

Executive Order 12333, *United States Intelligence Activities*, 4 December 1981.

Freeman, R. L., *Telecommunication System Engineering*, second edition, John Wiley & Sons, Inc., 1989.

GAMS Language Guide, release 2.25, GAMS Development Corporation, 1997.

Giarratano, J. C., *CLIPS User's Guide*, version 6.05, 1 November 1997.

Mitchell, B., *Utilization of the U.S. Telephone Network*, RAND, 1994.

Nemhauser, G., and Wolsey, L., *Integer and Combinatorial Optimization*, John Wiley & Sons, 1988.

Noll, A. M., *Introduction to Telephones and Telephone Systems*, second edition, Artech House, Inc., 1991.

Pearce, J., *Telecommunications Switching*, Plenum Press, New York, 1981.

Winston, W., *Operations Research, Applications and Algorithms*, third edition, International Thompson Publishing, 1994.

INITIAL DISTRIBUTION LIST

No. Copies

1. Defense Technical Information Center.....2
   8725 John J. Kingman Road, STE 0944
   Ft. Belvoir, VA 22060-6218

2. Dudley Knox Library.....2
   Naval Postgraduate School
   411 Dyer Road
   Monterey, CA 93943-5101

3. Director, Training and Education.....1
   MCCDC, Code C46
   1019 Elliot Rd.
   Quantico, VA 22134-5027

4. Director, Marine Corps Research Center.....2
   MCCDC, Code C40RC
   2040 Broadway Street
   Quantico, VA 33134-5107

5. Director, Studies and Analysis Division.....1
   MCCDC, Code C45
   300 Russell Road
   Quantico, VA 22134-5130

6. Marine Corps Representative.....1
   Naval Postgraduate School
   Code 037, Bldg. 234, HA-220
   699 Dyer Road
   Monterey, CA 93940

7. Marine Corps Tactical Systems Support Activity.....1
   Technical Advisory Branch
   Attn: Maj J. C. Cummiskey
   Box 555171
   Camp Pendleton, CA 92055-5080

8. Professor R. Kevin Wood, Code OR/Wd.....2
Department of Operations Research
Naval Postgraduate School
Monterey, CA 93943

9. Professor Gerald G. Brown, Code OR/Wd.....1
Department of Operations Research
Naval Postgraduate School
Monterey, CA 93943

10. National Security Agency.....2
Center for Operations Research
9800 Savage Rd
Fort Meade, MD 20755-6678

11. Allen S. Olson, Major, USMC.....2
Director, USA TRAC WSMR
White Sands Missile Range, NM 88002-5502